Blind Delegation with Certified Deletion

James Bartusek[*] Sanjam Garg[†] Dakshita Khurana[‡] Bhaskar Roberts[§]

Abstract

Can we outsource computation on encrypted sensitive data, while ensuring that the data is *certifiably, information-theoretically deleted* by the server after computation? This question was first posed by Broadbent and Islam (TCC 2020), and partial answers were recently provided by Poremba (arXiv 2022) and Bartusek and Khurana (arXiv 2022).

We present the first fully-secure solution to this problem of *blind delegation with certified deletion*. In more detail, we obtain maliciously-secure blind delegation for classical circuits, assuming post-quantum hardness of the learning with errors (LWE) problem. Our construction carefully combines classical fully-homomorphic encryption and succinct non-interactive arguments for P with quantum *subspace coset states*.

Building on these techniques, we also describe a generic compiler that adds a *publicly-verifiable* certified deletion property to a variety of cryptosystems, including blind delegation. Our publicly-verifiable schemes assume the existence of (post-quantum) indistinguishability obfuscation.

[*]UC Berkeley. Email: `bartusek.james@gmail.com`

[†]UC Berkeley and NTT Research. Email: `sanjamg@berkeley.edu`

[‡]UIUC. Email: `dakshita@illinois.edu`

[§]UC Berkeley. Email: `bhaskarr@berkeley.edu`

Contents

1 Introduction

Fully-homomorphic encryption facilitates impressive capabilities, most notably the ability to delegate computationally intensive tasks to untrusted servers while preserving the privacy of sensitive data.

In this setting, classical cryptosystems can only promise *computational* privacy. The untrusted server necessarily obtains a classical encoding that information-theoretically contains the sensitive data, and the only thing preventing recovery of all sensitive data is the conjectured hardness of a mathematical problem. If this problem becomes easy to solve in the future due to computational or scientific advances, or if the client's secret key is leaked to the server, there is no way to prevent the server from recovering the underlying plaintext.

On the other hand, recent advances in *certified deletion* [BI20, HMNY21, HMNY22b, Por22, BK22, HMNY22a] have demonstrated how *quantum information* can help get around this issue. In particular, some recent works [BI20, Por22, BK22] raise and explore the possibility of blind delegation with certified deletion. In its strongest form, this primitive will allow a weak client to delegate computations to a completely untrusted powerful server, while (i) keeping data computationally hidden from the server during protocol execution, and (ii) verifying that the server information-theoretically deletes this data afterwards.

This work gives the first *fully-secure* solution to the problem of blind delegation with certified deletion. Before stating these results in more detail, we introduce our definition of fully-secure blind delegation with certified deletion. Blind delegation is an interactive protocol that occurs between a computationally weak client with private input m and a computationally powerful server. The client requests that the server help them compute f for some computationally intensive function f, and at the end of the protocol the client should learn $f(m)$. In this work, we require that the following security properties hold against any quantum polynomial-time (QPT) malicious server.

- *Integrity.* The server cannot cause the client to accept an output $y \neq f(m)$.
- *Computational Privacy.* The server learns no information about the client's input m during the course of the protocol.
- *Certified Everlasting Privacy.* If the server produces a valid "certificate of deletion" at the end of the protocol, then the server's left-over state contains *information-theoretically* no information about m, *even* given the client's secret parameters.

We refer to a scheme that satisfies the above properties as a *fully* (or maliciously) secure blind delegation protocol with certified deletion. Next, we describe our results.

- **Fully secure blind delegation with certified deletion.** We develop the first fully secure blind delegation scheme with certified deletion, based on the quantum hardness of learning with errors (LWE). Our construction carefully combines the two classical LWE-based cryptographic primitives of compact fully-homomorphic encryption (FHE) and succinct non-interactive arguments (SNARGs) for P with quantum *subspace coset states*, which were introduced in [VZ21, CLLZ21].
- **Certified deletion with public verification.** We also develop a generic compiler that relies on subspace coset states to obtain a variety of primitives with *publicly* verifiable certified deletion.
  - We upgrade our construction of fully secure blind delegation with certified deletion to enable public verification of the deletion certificate.
  - In addition, for $X \in \{$public-key, attribute-based, witness, timed-release$\}$, we obtain post-quantum X encryption with *publicly-verifiable* certified deletion, assuming post-quantum X encryption.

  All our schemes satisfying publicly-verifiable certified deletion additionally assume *subspace-hiding obfuscation*, which is known from post-quantum indistinguishability obfuscation [Zha19].

1.1 Prior Work

1.1.1 Blind Delegation

As mentioned above, the recent works of [Por22] and [BK22] have also studied the primitive of blind delegation with certified deletion. We provide a comparison of our results with those of [Por22] and [BK22] in terms of both *security properties* and *computational assumptions.*

Security. Neither of the protocols from [Por22] or [BK22] are fully secure according to our definition. In particular, a malicious server (one who potentially deviates from the description of the protocol) can completely violate computational privacy, learning information about the client's input.

To see how, we first give a high-level description of the structure of the protocols, which both consist of four messages. First, the client encrypts their input m and sends a quantum ciphertext $|\mathsf{Enc}(m)\rangle$ to the server. Next, the server evaluates a function f to obtain a register holding a superposition over output ciphertexts $|\mathsf{Enc}(f(m))\rangle$, which is sent to the client. The client then coherently applies FHE decryption using their secret key, which allows them to recover $f(m)$ without disturbing the state, and then reverse their computation and send the undisturbed register back to the server. Finally, the server can uncompute f and recover the original ciphertext $|\mathsf{Enc}(m)\rangle$. Then, if they want, they can measure the ciphertext in a particular way to recover a certificate of deletion, which is sent to the client.

Now, consider the following attack. Suppose that the server wants to learn the first bit m_1 of m. They can easily prepare a state of the form

$$\frac{1}{\sqrt{2}}\,|\mathsf{Enc}(m_1)\rangle^{\mathsf{C}}\,|0\rangle^{\mathsf{S}} + \frac{1}{\sqrt{2}}\,|\mathsf{Enc}(0)\rangle^{\mathsf{C}}\,|1\rangle^{\mathsf{S}}\,,$$

where $\mathsf{Enc}(0)$ is a freshly prepared encryption of 0. Then, suppose they send register C to the client in place of the second message of the protocol described above. In the case that $m_1 = 0$, the client's computation will not disturb the state, and the server will receive back the C register unharmed. But in the case when $m_1 = 1$, the client's measurement of the output *will* collapse the state. Thus, if the server unentangles register C and S, measures S in the Hadamard basis, and observes outcome $|-\rangle$, they will learn *for sure* that $m_1 = 1$, breaking standard privacy of the protocol.

While this attack intuitively completely breaks the privacy of the protocol, we remark that it does not contradict any claims made in [Por22] or [BK22]. Indeed, neither [Por22] nor [BK22] claim that their protocol is "fully-secure" in the sense that we described above. In [Por22], the correctness and security properties are defined entirely separately. That is, it is argued that correctness of the four-message protocol holds assuming parties are honest, but *security* (and certified deletion security) is only argued *assuming that the server does not interact with the client*. Thus, the claim is essentially that *either* correctness of delegation holds, *or* privacy against a malicious server holds. But it is never claimed that both can hold simultaneously. In [BK22], the authors do jointly consider correctness and security of the four-message protocol. However, they only claim security against servers that are *semi-honest* during the protocol execution (and potentially malicious after, while producing the deletion certificate). Thus, the above attack is explicitly disallowed by the semi-honest assumption.

In this work, we provide the first fully-secure solution to blind delegation with certified deletion, explicitly preventing attacks like the one described above. Thus, our work is the first to establish the feasibility of blind delegation with certified deletion that can *simultaneously* provide correctness and security against potentially malicious servers.

We finally note that, unlike [Por22] and some other previous work on certified deletion [HMNY21, HMNY22b], our protocol promises *everlasting security* against an adversary that passes deletion verification (and receives the client's secret parameters). This is strictly stronger than previous definitions from [HMNY21, HMNY22b, Por22] that only promise security against *computationally-bounded* adversaries who receive the client's secret parameters.

Assumptions. Our protocol is proven secure under the sole assumption of the quantum hardness of LWE. Thus, we directly improve upon [BK22], whose protocol is also proven secure under LWE, but with the weaker semi-honest security property described above. This is also an improvement over [Por22], whose protocol is not reduced to a standard assumption. Instead, the security of this protocol relies on the conjecture that a particular hash function satisfies a newly introduced "strong Gaussian-collapsing" property.

Finally, as mentioned above, we also obtain blind delegation with *publicly-verifiable* certified deletion, under the additional assumption of post-quantum indistinguishability obfuscation. The protocol of [Por22] is also shown to be publicly-verifiable, under the strong Gaussian-collapsing conjecture, and thus our results on public verification are technically incomparable. However, we remark that [Por22]'s conjecture involves an interactive game that has a baked in certified deletion component, wherein the adversary receives some trapdoor information conditioned on them successfully returning a pre-image of the hash function. On the other hand, indistinguishability obfuscation a priori has nothing to do with certified deletion. While post-quantum indistinguishability obfuscation is also not known from standard assumptions, this is a very active area of research with many candidates proposed over the last few years [BGMZ18, CVW18, BDGM22, GP21, WW21, DQV+21].

1.1.2 Subspace coset states

We also mention the prior works that build cryptographic schemes from subspace coset states. These states were first used by [VZ21] in the context of proofs of quantum knowledge and by [CLLZ21] to construct signature tokens (among other unclonable primitives) in the plain model. Most recently, [AKL+] used subspace coset states to construct unclonable encryption satisfying the notion of unclonable indistinguishability. We remark that, while there are clearly similarities between the notions of unclonable encryption and encryption with certified deletion, our security definitions and proofs are quite different than those in [AKL+]. For example, [AKL+] crucially rely on *random oracles*, while our results are all in the plain model. Moreover, we achieve security definitions that promise *everlasting security* against unbounded adversaries after deletion, while [AKL+] focuses on computationally-bounded (or query-bounded) adversaries.

1.2 Techniques

Construction. The starting point for our blind delegation protocol is the recent protocol of [BK22]. Their construction relies on a classical fully-homomorphic encryption (FHE) scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ and BB84 states, which we describe using the notation $|x\rangle_\theta$, where $x \in \{0,1\}^n$ is a string of bits, and $\theta \in \{0,1\}^n$ is a string of basis choices.

- The client samples keys $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$ for the FHE scheme. Then, they encrypt their input $m \in \{0,1\}^D$ bit-by-bit as follows. For each bit m_i, sample n random BB84 states $|x_i\rangle_{\theta_i}$, and send

$$|x_i\rangle_{\theta_i}, \mathsf{ct}_i := \mathsf{Enc}\left(\mathsf{pk}, \left(\theta_i, m_i \oplus \bigoplus_{j:\theta_{i,j}=0} x_{i,j}\right)\right)$$

  to the server.

- The server first uses homomorphic evaluation in superposition to compute for each i

$$|x_i\rangle_{\theta_i}, \mathsf{ct}_i \rightarrow \sum |\mathsf{Enc}(m_i)\rangle,$$

  where the final summation represents a superposition over ciphertexts encrypting m_i, each with different random coins. Then, they homomorphically evaluate a circuit C of the client's choice in superposition to obtain

$$\sum |\mathsf{Enc}(m_1)\rangle, \ldots, \sum |\mathsf{Enc}(m_D)\rangle \rightarrow \sum |\mathsf{Enc}(C(m))\rangle,$$

  and they send the register holding the superposition over ciphertexts $\mathsf{Enc}(C(m))$ to the client.

- The client applies $\mathsf{Dec}(\mathsf{sk}, \cdot)$ in superposition, measures the output $C(m)$, reverses their $\mathsf{Dec}(\mathsf{sk}, \cdot)$ computation, and finally returns the register to the server.
- The server reverses their computation to end up with $|x_1\rangle_{\theta_1}, \mathsf{ct}_1, \dots, |x_D\rangle_{\theta_D}, \mathsf{ct}_D$. Finally, they can measure their BB84 states in the Hadamard basis to produce deletion certificates $(x'_1, \dots, x'_D)$. Each x'_i is accepted by the client if $x'_{i,j} = x_{i,j}$ for all j such that $\theta_{i,j} = 1$.

As described above, there is a very simple way for a malicious server to potentially learn a bit of the client's input, by preparing a state

$$\frac{1}{\sqrt{2}} |\mathsf{Enc}(m_i)\rangle^{\mathsf{C}} |0\rangle^{\mathsf{S}} + \frac{1}{\sqrt{2}} |\mathsf{Enc}(0)\rangle^{\mathsf{C}} |1\rangle^{\mathsf{S}},$$

and sending register C to the client. After the client returns register C, the server can implement a measurement that determines (with constant probability) whether register S was collapsed by the client's computation, which in turn indicates whether m_i is 0 or 1.

This attack relies on the fact that the client always immediately applies an operation that depends on their FHE secret key sk. To prevent this attack, we must introduce a way for the client to *check* that the server is honestly following the protocol, *before* using its secret key to operate on the state. Our solution involves two steps.

1. In the classical setting, a natural idea would be to have the server *prove* using a succinct non-interactive argument (SNARG) that they applied the correct FHE evaluation procedure to the input ciphertext. In our setting, this is not exactly feasible, since the input ciphertext is a quantum state. However, we observe that the server's computation is essentially just applying a classical functionality in superposition. That is, if we write our BB84 states as

   $$|x\rangle_\theta \propto \sum_{y:\{y_j = x_j\}_{j:\theta_j=0}} (-1)^{\bigoplus_{j:\theta_j=1} x_j \cdot y_j} |y\rangle,$$

   then the server's computation can be seen as applying a classical FHE evaluation to $\mathsf{ct}_1, \dots, \mathsf{ct}_D$, where the description of the functionality applied depends on the vectors y in each BB84 state.

   Thus, we include a SNARG in our scheme, and ask that the server augment their computation of $\mathsf{Enc}(C(m))$ with a SNARG proof π of correct evaluation (in superposition). The client will first verify the SNARG proof in superposition before applying their FHE decryption functionality. Since the SNARG instance now includes the vectors encoded in the BB84 states, the server must include these registers along with their register containing $\sum |\mathsf{Enc}(C(m)), \pi\rangle$, and the client will verify the SNARG proof with respect to the BB84 state registers.

   Finally, we remark that we only require SNARGs for P computation (as opposed to NP), and thus we can rely on a recent construction of SNARGs for P from the LWE assumption [CJJ21]. It is straightforward to verify that this SNARG remains post-quantum secure assuming the post-quantum hardness of LWE.

2. However, one issue remains. There is nothing preventing the server from using incorrect BB84 states. That is, the server is free to use any vectors y of their choice. This is a real problem, for a few reasons. First, a malicious server could clearly violate *integrity*, since using arbitrary y could result in the client accepting a computation of $C(m')$ for some arbitrary $m' \neq m$. Moreover, if the server can repeatedly query the client, they could launch an attack that allowed them to recover some θ_i, by preparing superpositions over honest evaluated ciphertexts and evaluated ciphertexts computed from BB84 states with a particular bit flipped (say, the j'th bit of the i'th set of states). If $C(m) \neq C(m')$, where m' is m with bit i flipped, then measuring whether the client's computation collapsed the server's register would determine whether $\theta_{i,j} = 0$ or $\theta_{i,j} = 1$. This would break certified everlasting privacy. Finally, one could even derive examples of functionalities C where, even with a single malicious query the

server could potentially learn some bit of m, by again preparing a superposition over honest ciphter-texts and "bit-flipped" ciphertexts, and measuring whether the client's computation collapsed their register or not. This would break standard privacy.

To remedy this issue, a natural idea is to have the client remember a description of their BB84 states, and simply check whether the server computed the SNARG with respect to these honest states. Unfortunately, this introduces another host of problems. First of all, the server can now use the client as an oracle that implements the description of their BB84 states. It is easy to see that they could learn a description of the states given such an oracle, and thus break certified everlasting privacy. In fact, it has even been shown how to do this without the client ever noticing that the server was behaving dishonestly [NSBU16]! Moreover, even if we explicitly disallow the server to interact with the client more than once, we will still not be able to satisfy our definition of certified everlasting privacy, which allows the server to obtain access to the client's secret parameters after deletion. If these parameters contain the FHE secret key along with a description of the BB84 states (which in particular includes the x_i), this allows easy recovery of m.

Our solution is to use *subspace coset states* rather than BB84 states. Subspace coset states are a generalization of BB84 states, and are defined as follows. Let S be a subspace of $\mathbb{F}_2^n$, let $\mathsf{co}(S)$ be a subspace of coset representatives of S, and let $\mathsf{co}(S^\perp)$ be a subspace of coset representatives of $S^\perp$. Then for $\mathsf{v} \in \mathsf{co}(S)$ and $\mathsf{w} \in \mathsf{co}(S^\perp)$, we define

$$|S_{\mathsf{v},\mathsf{w}}\rangle \coloneqq \frac{1}{\sqrt{2^n}} \sum_{\mathsf{s} \in S} (-1)^{\mathsf{s}\cdot\mathsf{w}} |\mathsf{s}+\mathsf{v}\rangle \, .$$

We will encrypt the plaintext using v, and consider any $\mathsf{z} \in S^\perp + \mathsf{w}$ (which can be obtained by measuring in the Hadamard basis) to be a valid deletion certificate.

Unlike BB84 states, which can be seen as subspace coset states where S is an "axis-aligned" subspace (spanned by a set of standard basis vectors), subspace coset states are highly entangled. This extra randomness actually solves the above issues: the server cannot learn a description of S given just oracle access to $S + \mathsf{v}$. Moreover, we can even have the client use a random *superspace* $T + \mathsf{u}$ of $S + \mathsf{v}$ for its check, and still argue security against a malicious server. The point here is that even if $T + \mathsf{u}$ is leaked after successful deletion, we can still argue that the client's plaintext m remains information-theoretically hidden.

As a bonus, using subspace coset states enables us to make certified deletion a *publicly-verifiable* process. As we describe in more detail at the end of this section, we can now release an obfuscated program that implements the (classical) functionality that checks for membership in $S^\perp + \mathsf{w}$, and still argue certified deletion security.

In summary, our blind delegation with certified deletion protocol follows [BK22], but with a couple of crucial changes: We replace the use of BB84 states with subspace coset states, and we request that the server applies a (post-quantum) SNARG in superposition.

Main theorem. Now, it remains to prove that our scheme satisfies certified everlasting hiding. That is, conditioned on the server producing vectors in the correct cosets $S_i^\perp + \mathsf{w}_i$, do they lose all information about m? By appealing to the soundness of SNARGs, we can reduce to the following generic certified deletion theorem.

Let Enc be a semantically secure encryption scheme. Suppose an adversary obtains

$$|S_{\mathsf{v},\mathsf{w}}\rangle \, , \mathsf{Enc}\left(S, b \oplus \bigoplus_{i \in [n]} \mathsf{v}_i\right) ,$$

where S is a random $n/2$-dimension subspace of $\mathbb{F}_2^n$, $\mathsf{v} \leftarrow \mathsf{co}(S)$, $\mathsf{w} \leftarrow \mathsf{co}(S^\perp)$, and v_i denotes the i^{th} bit of v. We claim that, if the adversary produces a vector $\mathsf{z} \in S^\perp + \mathsf{w}$, then they have information-theoretically lost all information about the bit b.

Technically, this claim should be strictly easier to prove than the corresponding claim in [BK22], which considers the case where $|S_{\mathsf{v},\mathsf{w}}\rangle$ are restricted to be BB84 states (in order words, when S is an axis-aligned subspace). However, we will eventually want to prove the stronger claim that the above holds even if the adversary is given oracle (or obfuscated) access to $S^\perp + \mathsf{w}$, and even if the adversary is given a random superspace $T + \mathsf{u}$ *in the clear*. We thus derive a significantly different proof from [BK22] that takes crucial advantage of the extra randomness in S. Our novel proof strategy is simultaneously simpler (in some sense) and more general than that of [BK22], and it gives rise to our new applications of maliciously-secure blind delegation and publicly-verifiable deletion.

Proof strategy. Our first steps mirror the proof of the main theorem in [BK22]. That is, we delay the dependence of the experiment on b by (i) replacing b in the encryption with a uniformly random bit b', and (ii) delaying the sampling of v until *after* the adversary has produced z. Then, we abort the experiment if $b' \neq b \oplus \bigoplus_{i\in[n]} \mathsf{v}_i$. Since the experiment now only depends on b in this final step, it suffices to show that, conditioned on the adversary producing a valid deletion certificate ($\mathsf{z} \in S^\perp + \mathsf{w}$), there is sufficient entropy in v, even conditioned on the adversary's state. That is, it suffices to show the following.

1. Sample $S, \mathsf{w} \leftarrow \mathsf{co}(S^\perp)$, and prepare the state
$$\frac{1}{\sqrt{|\mathsf{co}(S)|}} \sum_{\mathsf{v}\in\mathsf{co}(S)} |\mathsf{v}\rangle^{\mathsf{C}} |S_{\mathsf{v},\mathsf{w}}\rangle^{\mathsf{A}},$$
where C denotes the challenger's register and A denotes the adversary's register.

2. Give register A and $\mathsf{Enc}(S)$ to the adversary, who returns z.

3. Claim: If $\mathsf{z} \in S^\perp + \mathsf{w}$, then measuring register C results in a uniformly random $\mathsf{v} \leftarrow \mathsf{co}(S)$, sampled independently of register A.

Note that if the adversary could break Enc and learn S, they could use their knowledge of S to learn w and then measure their register to collapse v, falsifying the above Claim. Thus, we must at some point use the semantic security of Enc. To do so, we will define a *predicate* on the challenger's state, and show that (i) if this predicate holds, then the Claim above is true, (ii) by the semantic security of Enc, this predicate must hold regardless of whether we encrypt S or not, and (iii) the predicate holds information-theoretically when the adversary receives no information about S.

To define the predicate, we use intuition from the case where S is axis-aligned. In that case, S corresponds to $\theta \in \{0,1\}^n$, determining which qubits of $|S_{\mathsf{v},\mathsf{w}}\rangle$ are in the standard basis and which are in the Hadamard basis. Moreover, v is an n-bit string that can only potentially be non-zero on indices i such that $\theta_i = 0$, and w is an n-bit string that can only potentially be non-zero on indices i such that $\theta_i = 1$.

We expect that if the adversary returns $\mathsf{z} \in \{0,1\}^n$ such that $\{\mathsf{z}_i = \mathsf{w}_i\}_{i:\theta_i=1}$, then it must have measured (almost) all qubits of A in the Hadamard basis. Suppose that they actually *did* measure all their qubits in the Hadamard basis to produce the string z. Then, since C was entangled with A on qubits i such that $\theta_i = 0$ (where v is potentially non-zero), measuring these qubits of register C in the Hadamard basis would result in exactly $\{\mathsf{z}_i\}_{i:\theta_i=0}$. So suppose we applied a "Hadamard-and-shift" unitary $U_{\theta,\mathsf{w}}$ to C, which applies Hadamard to qubits i such that $\theta_i = 0$, and then applies the shift w to the resulting state. Then, the resulting state will be exactly the computational basis state $|\mathsf{z}\rangle$.

Now, we generalize this intuition to the case where S is an arbitrary subspace. Here, our "Hadamard-and-shift" unitary $U_{S,\mathsf{w}}$ would be the map[1]

[1]Technically this is a unitary transformation over an $2^{n/2}$-dimensional Hilbert space. That is, we can define maps $e : \{0,1\}^{n/2} \to \mathsf{co}(S)$ and $f : \{0,1\}^{n/2} \to S^\perp + \mathsf{w}$, and define, for any $t \in \{0,1\}^{n/2}$, $U_{S,\mathsf{w}} |t\rangle \to \sum_{t'\in\{0,1\}^{n/2}} (-1)^{e(t)\cdot f(t')} |t'\rangle$. Though in the body, we actually define $U_{S,\mathsf{w}}$ as an arbitrary 2^n-dimensional "completion" of the $2^{n/2}$-dimensional unitary just described.

$$U_{S,\mathsf{w}} : \mathsf{co}(S) \to S^\perp + \mathsf{w}, \quad \text{where for } \mathsf{v} \in \mathsf{co}(S), \quad U_{S,\mathsf{w}} |\mathsf{v}\rangle \to \sum_{\widetilde{\mathsf{v}} \in S^\perp + \mathsf{w}} (-1)^{\mathsf{v} \cdot \widetilde{\mathsf{v}}} |\widetilde{\mathsf{v}}\rangle .$$

If the adversary measures all qubits of A in the Hadamard basis to obtain z, then applying $U_{S,\mathsf{w}}$ to register C would result in the computational basis state $|\mathsf{z}\rangle$. Of course, an adversary might not simply be measuring their register in the Hadamard basis. Nevertheless, we claim the following:

Conditioned on $\mathsf{z} \in S^\perp + \mathsf{w}$ *in the above experiment, it holds that applying* $U_{S,\mathsf{w}}$ *to register* C *results in the state* $|\mathsf{z}\rangle$ *(up to negligible trace distance).*

A couple of remarks are in order.

- For any $\mathsf{z} \in S^\perp + \mathsf{w}$, applying $U_{S,\mathsf{w}}^\dagger$ to $|\mathsf{z}\rangle$ results in a uniform superposition over $\mathsf{v} \in \mathsf{co}(S)$. We show in the body that (with overwhelming probability over the sampling of S), the bit $\bigoplus_{i \in [n]} \mathsf{v}_i$ is uniformly random when $\mathsf{v} \leftarrow \mathsf{co}(S)$. Thus, to complete the proof of our main theorem, it suffices to prove this statement.
- In the BB84 state / axis-aligned case, as considered by [BK22], we cannot hope to prove such a strong statement. Indeed, the predicate defined in [BK22] was more complicated, and resulted in the need for a "quantum cut-and-choose" analysis [BF10]. Thus, moving to more general subspace coset states simplifies our overall proof strategy.

Finally, it remains to prove this statement. This will follow by showing that the adversary can only win the following game with negligible probability.

1. Sample $S, \mathsf{w} \leftarrow \mathsf{co}(S^\perp)$, prepare the state

$$\frac{1}{\sqrt{|\mathsf{co}(S)|}} \sum_{\mathsf{v} \in \mathsf{co}(S)} |\mathsf{v}\rangle^{\mathsf{C}} |S_{\mathsf{v},\mathsf{w}}\rangle^{\mathsf{A}} ,$$

   and then "pre-emptively" apply $U_{S,\mathsf{w}}$ to C and measure in the standard basis to obtain a vector $\widetilde{\mathsf{z}}$. Following the intuition given above, it can be shown that the state on register A collapses to $H^{\otimes n} |\widetilde{\mathsf{z}}\rangle$.
2. Give $H^{\otimes n} |\widetilde{\mathsf{z}}\rangle$ and $\mathsf{Enc}(S)$ to the adversary, who returns z.
3. The adversary wins if $\mathsf{z} \in S^\perp + \mathsf{w}$ but $\mathsf{z} \neq \widetilde{\mathsf{z}}$.

By semantic security, we can remove the encryption of S. We can also strip off $H^{\otimes n}$, resulting in the following very simple game: Sample $S, \mathsf{w} \leftarrow \mathsf{co}(S^\perp), \widetilde{\mathsf{z}} \leftarrow S^\perp + \mathsf{w}$ and give $\widetilde{\mathsf{z}}$ to the adversary. The adversary wins if it can produce a different vector z in the same coset of $S^\perp$. Since the adversary receives no information about S, it is straightforward to show that this occurs with negligible probability. However, we remark again that in the BB84 state case, this probability would *not* be negligible, since the adversary could win with constant probability by flipping a single bit of $\widetilde{\mathsf{z}}$. So, this step is where we rely on the fact that we are using general subspace coset states.

Extensions. We conclude our overview by discussing a couple of ways that we extend the above proof.

First, we consider the question of publicly-verifiable deletion. The deletion verification procedure simply checks whether the certificate z is in the coset $S^\perp + \mathsf{w}$. This can be made public by giving out an *obfuscated program* for membership in $S^\perp + \mathsf{w}$. If S is axis-aligned, then this would completely compromise security, since a description of S can easily be learned using black-box access to $S^\perp + \mathsf{w}$. On the other hand, this is not necessarily true when S is a uniformly random subspace. In fact, [Zha19] showed that if $\mathsf{i}\mathcal{O}$ is a post-quantum indistinguishability obfuscator, then no adversary can distinguish whether they are given $\mathsf{i}\mathcal{O}(S)$ or $\mathsf{i}\mathcal{O}(T)$ for a random superspace T of S. Thus, even if the adversary is additionally given $\mathsf{i}\mathcal{O}(S^\perp + \mathsf{w})$, then (during the final sequence of steps described above) we can replace $\mathsf{i}\mathcal{O}(S^\perp + \mathsf{w})$ with $\mathsf{i}\mathcal{O}(T + \mathsf{w})$ for a large superspace T of $S^\perp$. Then, we can still conclude that the adversary, given $\widetilde{z} \in S^\perp + \mathsf{w}$, does not have sufficient information about S to guess any $\mathsf{z} \in S^\perp + \mathsf{w}$ such that $\mathsf{z} \neq \widetilde{\mathsf{z}}$.

Next, we recall that during our blind delegation with certified deletion protocol, the client has to check that the server is honestly using vectors in $S + \mathsf{v}$ for their computation. In other words, the client must keep $S + \mathsf{v}$ around as part of their secret state. But this means that if the client's state is leaked, we have no hope of obtaining certified everlasting security, since the server has a classical encryption of S and a classical encryption of the client's input masked by v. To remedy this, we will actually have the client sample a random larger affine space $T + \mathsf{u}$ such that $S + \mathsf{v} \subset T + \mathsf{u}$, and use $T + \mathsf{u}$ for this check rather than $S + \mathsf{v}$. We can show that, even if the adversary receives $T + \mathsf{u}$ *in the clear*, then the above certified deletion theorem still holds. This follows essentially by treating $T + \mathsf{u}$ as the ambient space, and we defer details of this to the proof of our main theorem in the body.

2 Preliminaries

Let λ denote the security parameter. A function f is **negligible** if for every constant $c \in \mathbb{N}$, $f(n) < n^{-c}$ for sufficiently large n. We use $\mathrm{negl}(n)$ to represent a generic negligible function.

A **quantum register** X is a set of n qubits, which represent the Hilbert space $\mathbb{C}^{2^n}$ and hold a quantum state. A **quantum state** on register X is represented as ρ or ρ^{X}, and it is a positive semi-definite operator on $\mathbb{C}^{2^n}$ with trace 1. We sometimes refer to this PSD operator as a **density matrix**. If the state is **pure**, it can also be represented as a unit vector $|\psi\rangle \in \mathbb{C}^{2^n}$.

A **quantum operation** F maps a state ρ on register X to a state σ on register Y. Formally, a quantum operation is a completely-positive trace-preserving (CPTP) map. Note that the two registers may have different sizes. Also, some possible notations for F include: $\sigma = F(\rho)$, $\sigma^{\mathsf{Y}} = F(\rho^{\mathsf{X}})$ and $\mathsf{Y} \leftarrow F(\mathsf{X})$.

A **unitary** $U : \mathsf{X} \to \mathsf{X}$ is a special case of a quantum operation that satisfies $U^\dagger U = UU^\dagger = \mathbb{I}^{\mathsf{X}}$, where $\mathbb{I}^{\mathsf{X}}$ is the identity matrix on register X. A **projector** Π is a Hermitian operator such that $\Pi^2 = \Pi$, and a **projective measurement** is a collection of projectors $\{\Pi_i\}_i$ such that $\sum_i \Pi_i = \mathbb{I}$.

Trace distance is the quantum analog of total variation distance: for two states ρ and σ, the trace distance between them is an upper bound on the advantage with which an (unbounded) algorithm can distinguish ρ and σ. Formally, the trace distance between ρ and σ is given by:

$$\mathsf{TD}(\rho, \sigma) := \frac{1}{2}\mathsf{Tr}\left(\sqrt{(\rho - \sigma)^\dagger(\rho - \sigma)}\right)$$

where $\mathsf{Tr}(\cdot)$ is the trace of the given matrix.

2.1 Subspaces and Cosets

The first set of facts come from [CLLZ21].

For any vectors $\mathsf{u}, \mathsf{v} \in \mathbb{F}_2^n$, let the **inner product** be computed mod 2: $\langle \mathsf{u}, \mathsf{v}\rangle = \sum_i \mathsf{u}_i \cdot \mathsf{v}_i \mod 2$. For a subspace $S \leq \mathbb{F}_2^n$, the **orthogonal subspace** is $S^\perp = \{\mathsf{v} \in \mathbb{F}_2^n : \langle \mathsf{u}, \mathsf{v}\rangle = 0, \forall \mathsf{u} \in S\}$. Their dimensions satisfy $\dim(S) + \dim(S^\perp) = n$.

For any vector $\mathsf{z} \in \mathbb{F}_2^n$, the **coset** of S that z belongs to is the set: $S + \mathsf{z} = \{\mathsf{v} \in \mathbb{F}_2^n | \mathsf{v} = \mathsf{s} + \mathsf{z}, \mathsf{s} \in S\}$. The cosets of S partition $\mathbb{F}_2^n$ into equal-sized sets: $|S + \mathsf{z}| = |S|$, and every z belongs to exactly one coset of S.

Finally, a **subspace coset state** is a state of the following form, where $\mathsf{v}, \mathsf{w} \in \mathbb{F}_2^n$:

$$|S_{\mathsf{v},\mathsf{w}}\rangle = \frac{1}{\sqrt{|S|}} \sum_{\mathsf{z} \in S + \mathsf{v}} |\mathsf{z}\rangle \cdot (-1)^{\langle \mathsf{z}, \mathsf{w}\rangle}$$

Also, $H^{\otimes n} |S_{\mathsf{v},\mathsf{w}}\rangle = |S^\perp_{\mathsf{w},\mathsf{v}}\rangle$, where H is the single-qubit Hadamard gate.

Now we'll introduce definitions of our own and prove some additional facts.

We will represent the cosets of S with a subspace $\mathsf{co}(S)$ that contains exactly one element from each coset. $\mathsf{co}(S)$ is useful for decomposing vectors since every $\mathsf{z} \in \mathbb{F}_2^n$ has a unique decomposition: $(\mathsf{u}, \mathsf{v}) \in S \times \mathsf{co}(S)$ such that $\mathsf{z} = \mathsf{u} + \mathsf{v}$ (Lemma 2.1).

It is useful intuition to imagine choosing $\mathsf{co}(S) = S^\perp$, but this is not always allowed. Because S is a subspace of $\mathbb{F}_2^n$ and not $\mathbb{R}^n$, it's possible that some coset of S contains multiple vectors from $S^\perp$ and another coset contains none.

There are usually many valid choices of $\mathsf{co}(S)$ for a given S. In this paper, we will choose $\mathsf{co}(S)$ using the sampling procedure given in Definition 2.2 below. Also, to avoid ambiguity, we always assume that a description of subspace S includes a basis for each of the following subspaces: $[S, S^\perp, \mathsf{co}(S), \mathsf{co}(S^\perp)]$.

Lemma 2.1 gives some useful properties of $\mathsf{co}(S)$.

Lemma 2.1. *For subspaces $S, \mathsf{co}(S) \leq \mathbb{F}_2^n$, the following are equivalent:*

1. *$\mathsf{co}(S)$ contains exactly one element from each coset of S.*
2. *For any $\mathsf{z} \in \mathbb{F}_2^n$, there is a unique pair $(\mathsf{u}, \mathsf{v}) \in S \times \mathsf{co}(S)$ such that $\mathsf{z} = \mathsf{u} + \mathsf{v}$.*
3. *$\dim[\mathsf{co}(S)] = n - \dim(S)$ and $\mathsf{span}[S, \mathsf{co}(S)] = \mathbb{F}_2^n$.*

Proof. First, (1) implies (2). For any $\mathsf{z} \in \mathbb{F}_2^n$, z belongs to a coset of S, and there is a unique $\mathsf{v} \in \mathsf{co}(S)$ from the same coset. Let $\mathsf{u} = \mathsf{z} - \mathsf{v}$. Since z, v belong to the same coset, $\mathsf{u} \in S$, and $\mathsf{z} = \mathsf{u} + \mathsf{v}$.

Second, (2) implies (3). (2) directly shows that $\mathsf{span}[S, \mathsf{co}(S)] = \mathbb{F}_2^n$. Next, we'll show that $\dim[\mathsf{co}(S)] = n - \dim(S)$. Every $\mathsf{z} \in \mathbb{F}_2^n$ has a *unique* decomposition in $S \times \mathsf{co}(S)$, so $|\mathbb{F}_2^n| = |S \times \mathsf{co}(S)|$. Therefore, $|\mathsf{co}(S)| = 2^n/|S|$, and $\dim[\mathsf{co}(S)] = n - \dim(S)$.

Third, (3) implies (1). $\mathbb{F}_2^n = \mathsf{span}[S, \mathsf{co}(S)]$, so for any vector $\mathsf{z} \in \mathbb{F}_2^n$, $\mathsf{z} = \mathsf{u} + \mathsf{v}$ for some $(\mathsf{u}, \mathsf{v}) \in S \times \mathsf{co}(S)$. Next, $\mathsf{v} = \mathsf{z} - \mathsf{u} \in S + \mathsf{z}$, so v is in both $\mathsf{co}(S)$ and $S + \mathsf{z}$. Therefore, every coset shares at least one element with $\mathsf{co}(S)$. Furthermore, $\mathsf{co}(S)$ contains exactly one vector from each coset because $|\mathsf{co}(S)| = 2^n/|S|$, which equals the number of cosets. $\square$

Sampling Procedure

The procedure in Definition 2.2 below describes how to sample the coset subspaces $\mathsf{co}(S)$ and $\mathsf{co}(S^\perp)$ for a given subspace S. Throughout the paper, we will actually need to sample two subspaces $S \leq T$ with additional properties, and the procedure in Definition 2.2 also meets these requirements.

Let us be given two dimensions $d_S, d_T \in \mathbb{N}$ such that $1 \leq d_S \leq d_T \leq n$. Our goal is to sample two subspaces S, T uniformly at random such that $\dim(S) = d_S$, and $\dim(T) = d_T$, and $S \leq T$. We will use the following procedure:

Definition 2.2 (Procedure to Sample S, T). *Sample n linearly independent vectors $\{\mathsf{z}_1, \ldots, \mathsf{z}_n\}$ uniformly at random. Then let*

$$\begin{aligned} S &= \mathsf{span}(\mathsf{z}_1, \ldots, \mathsf{z}_{d_S}) \\ \mathsf{co}(S) &= \mathsf{span}(\mathsf{z}_{d_S+1}, \ldots, \mathsf{z}_n) \\ \mathsf{co}(S^\perp) &= \mathsf{co}(S)^\perp \end{aligned}$$

$$\begin{aligned} T &= \mathsf{span}(\mathsf{z}_1, \ldots, \mathsf{z}_{d_T}) \\ \mathsf{co}(T) &= \mathsf{span}(\mathsf{z}_{d_T+1}, \ldots, \mathsf{z}_n) \\ \mathsf{co}(T^\perp) &= \mathsf{co}(T)^\perp \end{aligned}$$

Next, choose a new basis for each of the 8 subspaces, independently and uniformly at random. Note that the subspaces do not change – just the bases used to represent them.

Finally, the description of S comprises the bases for $[S, S^\perp, \mathsf{co}(S), \mathsf{co}(S^\perp)]$, and the description of T comprises the bases for $[T, T^\perp, \mathsf{co}(T), \mathsf{co}(T^\perp)]$.

The reason we choose fresh random bases for each subspace is so that someone with a description of T but not S does not learn anything about S, other than the fact that $S \leq T$. Originally, we sampled S and T using a common basis $\{\mathsf{z}_i\}_{i\in[n]}$. If we represented T as $(\mathsf{z}_1, \dots, \mathsf{z}_{d_T})$ and $\mathsf{co}(T)$ as $(\mathsf{z}_{d_T+1}, \dots, \mathsf{z}_n)$, then someone with those vectors could learn S.

The following lemma shows that Definition 2.2 is a valid way to sample S, T and the corresponding co subspaces.

Lemma 2.3. *The output of the sampling procedure given by Definition 2.2 has the following properties:*

1. *S, T are uniformly random subspaces such that $\dim(S) = d_S, \dim(T) = d_T$, and $S \leq T$.*
2. *$\mathsf{co}(S)$ contains exactly one element from each coset of S. The analogous statement holds for $\mathsf{co}(S^\perp), \mathsf{co}(T), \mathsf{co}(T^\perp)$.*
3. *$\mathsf{co}(T) \leq \mathsf{co}(S)$ and $\mathsf{co}(S^\perp) \leq \mathsf{co}(T^\perp)$*

Proof. Property 1 is clear from the sampling procedure. For property 2, we will show that $\dim[\mathsf{co}(S)] = n - \dim(S)$ and $\mathsf{span}[S, \mathsf{co}(S)] = \mathbb{F}_2^n$. By Lemma 2.1, this implies that $\mathsf{co}(S)$ contains exactly one element from each coset of S. We will show the analogous statements for $\mathsf{co}(S^\perp), \mathsf{co}(T), \mathsf{co}(T^\perp)$.

First, it is clear by the construction of $\mathsf{co}(S)$ that $\dim[\mathsf{co}(S)] = n - \dim(S)$. It is also clear that $\dim[\mathsf{co}(T)] = n - \dim(T)$. Next,

$$\dim[\mathsf{co}(S^\perp)] = \dim[\mathsf{co}(S)^\perp] = n - \dim[\mathsf{co}(S)] = n - [n - \dim(S)] = \dim(S) = n - \dim(S^\perp)$$

By the same argument, we can show that $\dim[\mathsf{co}(T^\perp)] = n - \dim(T^\perp)$.

Second,

$$\mathsf{span}[S, \mathsf{co}(S)] = \mathsf{span}(\mathsf{z}_1, \dots, \mathsf{z}_n) = \mathbb{F}_2^n$$

because the vectors $\{\mathsf{z}_i\}_{i\in[n]}$ are linearly independent. By the same reasoning, we can show that $\mathsf{span}[T, \mathsf{co}(T)] = \mathbb{F}_2^n$.

Next, we will prove that $\mathsf{span}[S^\perp, \mathsf{co}(S)^\perp] = \mathbb{F}_2^n$. First, $S \cap \mathsf{co}(S) = \{\mathbf{0}\}$. This is because $\mathsf{co}(S)$ contains exactly one element of S (Lemma 2.1), and the shared element has to be $\mathbf{0}$ because S and $\mathsf{co}(S)$ are both subspaces. Second:

$$\mathbb{F}_2^n = \{\mathbf{0}\}^\perp = [S \cap \mathsf{co}(S)]^\perp = \mathsf{span}[S^\perp, \mathsf{co}(S)^\perp]$$

For the same reason, $\mathsf{span}[T^\perp, \mathsf{co}(T)^\perp] = \mathbb{F}_2^n$.

Now we'll prove property 3. It is clear from the construction that $\mathsf{co}(T) \leq \mathsf{co}(S)$. Next,

$$\mathsf{co}(S^\perp) = \mathsf{co}(S)^\perp \leq \mathsf{co}(T)^\perp = \mathsf{co}(T^\perp)$$

□

2.2 Subspace-Hiding Obfuscation

For a subspace S and a vector $\mathsf{z} \in \mathbb{F}_2^n$, let the function P_S accept the vectors in S and reject all others. Also, let $P_{S+\mathsf{z}}$ accept the vectors in $S + \mathsf{z}$ and reject all others.

A **subspace-hiding obfuscator** (shO) is a program that obfuscates P_S. For security, we say that an adversary cannot distinguish $\mathsf{shO}(P_S)$ from the obfuscation of a random superspace of S.

Definition 2.4 ([Zha19], Def. 6.2)**.** *A subspace-hiding obfuscator (shO) for a field $\mathbb{F}$ and dimensions d_0, d_1 is a PPT algorithm shO that obfuscates P_S. shO is secure all QPT adversaries have negligible advantage in the following game:*

- *The adversary sends the challenger a subspace $S_0 \leq \mathbb{F}^n$ of dimension d_0.*

- *The challenger samples a random subspace $S_1 \leq \mathbb{F}^n$ of dimension d_1 such that $S_0 \leq S_1$. Then they sample $b \leftarrow \{0,1\}$ and compute $\hat{P} \leftarrow \mathsf{shO}(P_{S_b})$ and send $\hat{P}$ to the adversary.*
- *The adversary makes a guess b' for b.*

The adversary's advantage is $|\mathbb{P}(b' = b) - 1/2|$.

Theorem 2.5 ([Zha19], Theorem 6.3)**.** *If injective one-way functions exist and $|\mathbb{F}|^{n-d_1}$ is exponential, then any indistinguishability obfuscator, appropriately padded, is a subspace hiding obfuscator for field $\mathbb{F}$ and dimensions d_0, d_1.*

2.3 SNARGs for P

We define the notion of publicly-verifiable non-interactive delegation for a Turing machine $\mathcal{M}$. Such a scheme consists of algorithms $(\mathsf{SNARG.Gen}, \mathsf{SNARG.Prove}, \mathsf{SNARG.Verify})$ with the following syntax.

- $\mathsf{SNARG.Gen}(1^\lambda, \mathcal{M}, t, n) \rightarrow (\mathsf{pk}, \mathsf{vk})$ is an algorithm that takes as input a security parameter 1^λ, a Turing machine $\mathcal{M}$, a time bound t, and an input length n, and outputs a prover key pk and a verifier key vk.
- $\mathsf{SNARG.Prove}(\mathsf{pk}, x) \rightarrow (y, \pi)$ is an algorithm that takes as input a prover key pk and an input $x \in \{0,1\}^n$ and outputs $y \in \{0,1\}^m$ and a proof π.
- $\mathsf{SNARG.Verify}(\mathsf{vk}, x, y, \pi) \rightarrow \{0,1\}$ is an algorithm that takes as input a verifier key vk, an input x, an output y, and a proof π, and outputs either 0 or 1.

For Turing machine $\mathcal{M}$, let $\mathcal{U}_\mathcal{M}$ be the language

$$\mathcal{U}_\mathcal{M} \coloneqq \{(x, y, t) : \mathcal{M}(x) \text{ outputs } y \text{ within } t \text{ steps}\}.$$

Given a Turing machine $\mathcal{M}$ and time bound t, we also define $y \coloneqq \mathcal{M}_t(x)$ to be the value written on the output tape of the Turing machine after t steps, if such a value exists.

Definition 2.6. $(\mathsf{SNARG.Gen}, \mathsf{SNARG.Prove}, \mathsf{SNARG.Verify})$ *is a publicly-verifiable non-interactive delegation scheme for Turing machine $\mathcal{M}$ if the following hold.*

- ***Correctness:*** *For all λ, n, t such that $n \leq t \leq 2^\lambda$, and $(x, y, t) \in \mathcal{U}_\mathcal{M}$, it holds that*

$$\Pr\left[\mathsf{SNARG.Verify}(\mathsf{vk}, x, y, \pi) = 1 : \begin{array}{r} (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathsf{SNARG.Gen}(1^\lambda, \mathcal{M}, t, n) \\ (y, \pi) \leftarrow \mathsf{SNARG.Prove}(\mathsf{pk}, x) \end{array}\right] = 1.$$

- ***Efficiency:*** *In the above completeness experiment,* $\mathsf{SNARG.Prove}$ *runs in time* $\mathrm{poly}(n, t, \lambda)$ *and* $\mathsf{SNARG.Verify}$ *runs in time* $\mathrm{poly}(n, \log t, \lambda)$.
- ***Soundness:*** *For every QPT adversary $\{\mathcal{A}_\lambda\}_\lambda$ and polynomials $t(\lambda), n(\lambda)$, it holds that*

$$\Pr\left[\begin{array}{l} \mathsf{SNARG.Verify}(\mathsf{vk}, x, y, \pi) = 1 \\ (x, y, t) \notin \mathcal{U}_\mathcal{M} \end{array} : \begin{array}{r} (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathsf{SNARG.Gen}(1^\lambda, \mathcal{M}, t, n) \\ (x, y, \pi) \leftarrow \mathcal{A}_\lambda(\mathsf{pk}, \mathsf{vk}) \end{array}\right] = \mathrm{negl}(\lambda).$$

2.4 Fully-Homomorphic Encryption

A fully-homomorphic encryption scheme consists of algorithms $\mathsf{FHE} = (\mathsf{FHE.Gen}, \mathsf{FHE.Enc}, \mathsf{FHE.Eval}, \mathsf{FHE.Dec})$ with the following syntax.

- $\mathsf{FHE.Gen}(1^\lambda) \rightarrow (\mathsf{pk}, \mathsf{sk})$ is an algorithm that takes as input the security parameter and outputs a public key pk and a secret key sk.
- $\mathsf{FHE.Enc}(\mathsf{pk}, m) \rightarrow \mathsf{ct}$ is an algorithm that takes as input the public key pk and a message m, and outputs a ciphertext ct.

- $\mathsf{FHE.Eval}(\mathsf{pk}, C, \mathsf{ct}) \to \widetilde{\mathsf{ct}}$ is an algorithm that takes as input the public key pk, a circuit C, and a ciphertext ct, and outputs an evaluated ciphertext $\widetilde{\mathsf{ct}}$.
- $\mathsf{FHE.Dec}(\mathsf{sk}, \mathsf{ct}) \to m$ is an algorithm that takes as input the secret key sk and a ciphertext ct, and outputs a message m.

An FHE scheme should satisfy **semantic security**, which states that the encryptions of two equal-length strings m_0, m_1 are computationally indistinguishable, and **correctness of evaluation**, which states that for any input m and circuit C, $\mathsf{FHE.Dec}(\mathsf{sk}, \mathsf{FHE.Eval}(\mathsf{pk}, C, \mathsf{ct})) = C(m)$, where $\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, m)$. An FHE scheme should also satisfy some notion of **compactness**, which bounds the size of ciphertexts. In a levelled FHE scheme, the size of ct may grow with the depth of the circuit C to be evaluated, while in an unlevelled FHE scheme, the size of ct must be independent of C. Levelled FHE schemes are known from the hardness of the LWE, and unlevelled FHE schemes additionally require a circular-security assumption [Gen09, BV11, GSW13].

3 Definitions

Blind delegation is an interactive protocol that allows a client with limited computational resources to compute a resource-intensive function with the help of a more-powerful server. The client has an input x, a Turing machine $\mathcal{M}$, and a time bound t, and asks the server to help them compute $\mathcal{M}_t(x)$. For security, we require that a malicious server cannot force the client to output an incorrect answer for $\mathcal{M}_t(x)$, or learn any information about x.

Blind delegation can be achieved with fully-homomorphic encryption. In such a scheme, the server gets an encryption of x and computes the encryption of $\mathcal{M}_t(x)$ homomorphically. However, although x is computationally hidden from the server, if the decryption key is leaked later on, then the server can learn the message. Moreover, a computationally unbounded server can always eventually recover x.

Blind delegation with *certified deletion* provides a stronger guarantee. After computing $\mathcal{M}_t(x)$, the server deletes the ciphertext encrypting x and certifies that they have done so. From then on, x is statistically hidden from the server. We even require that if the client's decryption / verification keys are leaked, x is still statistically hidden from the server.

In this section, we'll define the syntax of blind delegation with certified deletion as well as the correctness and efficiency properties. Then, we present our notion of security, which captures the three properties of *Integrity*, *Computational Privacy*, and *Certified Everlasting Privacy*, which should hold against malicious servers. We refer to a blind delegation protocol that satisfies these security properties as *fully-secure*.

Definition 3.1 (Blind delegation with certified deletion). *Blind delegation with certified deletion is an interactive protocol between a server $\mathcal{S}$ and a client $\mathcal{C}$. It is composed of three sub-protocols,* Gen, Eval, *and* Del.

1. $\mathsf{Gen}(1^\lambda, \mathcal{M}, t, x) \to (\mathsf{pp}, \mathsf{sp}, \mathsf{vk}, |\mathsf{ct}\rangle)$*: The* Gen *algorithm is run by the client. It takes as input the security parameter 1^λ, a Turing machine $\mathcal{M}$, a time bound t, and an input x. It outputs public parameters* pp, *secret parameters* sp, *a (potentially secret) verification key* vk, *and a quantum ciphertext $|\mathsf{ct}\rangle$ on register* C.
2. $\mathsf{Eval}\langle\mathcal{C}(\mathsf{sp}), \mathcal{S}(\mathsf{pp}, \mathsf{C})\rangle \to (b_{\mathsf{OutVer}}, y, \mathsf{C})$*:* Eval *is an interactive protocol between the client with input* sp *and the server with input* pp *and a state on register* C. *At the end of the protocol, the client learns a bit $b_{\mathsf{OutVer}} \in \{\top, \bot\}$ which indicates whether the client has accepted an output and a string $y \in \{0,1\}^* \cup \{\bot\}$, and the server outputs a state on register* C.
3. $\mathsf{Del}\langle\mathcal{C}(\mathsf{vk}), \mathcal{S}(\mathsf{pp}, \mathsf{C})\rangle \to b_{\mathsf{DelVer}}$*:* Del *is an interactive protocol between the client with input* vk *and the server with input* pp *and a state on register* C. *At the end of the protocol, the client outputs a bit $b_{\mathsf{DelVer}} \in \{\top, \bot\}$.*

Definition 3.2 (Efficiency). *A protocol for blind delegation with certified deletion is **efficient** if the server runs in time $\mathrm{poly}(|x|, t, \lambda)$ and the client runs in time $\mathrm{poly}(|x|, \log t, \lambda)$.*

We require the client to be more efficient than the server because the client presumably doesn't have time to compute $\mathcal{M}_t$.

Definition 3.3 (Correctness). *A protocol for blind delegation with certified deletion is **correct** if for any Turing machine $\mathcal{M}$, time bound $t = t(\lambda)$, and input sequence $\{x_\lambda\}_{\lambda \in \mathbb{N}}$ such that for all λ, $\mathcal{M}(x_\lambda)$ halts within $t(\lambda)$ timesteps, it holds that*

$$\Pr\left[b_{\mathsf{OutVer}} = \top \wedge y = \mathcal{M}_t(x_\lambda) \wedge b_{\mathsf{DelVer}} = \top : \begin{array}{r} (\mathsf{pp}, \mathsf{sp}, \mathsf{vk}, |\mathsf{ct}\rangle) \leftarrow \mathsf{Gen}(1^\lambda, \mathcal{M}, t, x_\lambda) \\ (b_{\mathsf{OutVer}}, y, \mathsf{C}) \leftarrow \mathsf{Eval}\langle \mathcal{C}(\mathsf{sp}), \mathcal{S}(\mathsf{pp}, \mathsf{C})\rangle \\ b_{\mathsf{DelVer}} \leftarrow \mathsf{Del}\langle \mathcal{C}(\mathsf{vk}), \mathcal{S}(\mathsf{pp}, \mathsf{C})\rangle \end{array}\right] = 1 - \mathrm{negl}(\lambda).$$

Definition 3.4 (Security). *A protocol for blind delegation with certified deletion is **secure** if the following properties hold for any non-uniform QPT adversarial server $\mathcal{S}^* = \{\mathcal{S}^*_\lambda, |\psi_\lambda\rangle\}_{\lambda \in \mathbb{N}}$, where $\mathcal{S}^*_\lambda$ is an interactive quantum machine that maintains a state on register C^*, and $|\psi_\lambda\rangle$ is some polynomial-size non-uniform advice.*

- ***Integrity:*** *For any Turing machine $\mathcal{M}$, time bound $t = t(\lambda)$, and input sequence $\{x_\lambda\}_{\lambda \in \mathbb{N}}$ such that $\mathcal{M}(x_\lambda)$ halts within t timesteps, it holds that*

$$\Pr\left[b_{\mathsf{OutVer}} = \top \wedge y \neq \mathcal{M}_t(x_\lambda) : \begin{array}{r} (\mathsf{pp}, \mathsf{sp}, \mathsf{vk}, |\mathsf{ct}\rangle) \leftarrow \mathsf{Gen}(1^\lambda, \mathcal{M}, t, x_\lambda) \\ (b_{\mathsf{OutVer}}, y, \mathsf{C}^*) \leftarrow \mathsf{Eval}\langle \mathcal{C}(\mathsf{sp}), \mathcal{S}^*_\lambda(\mathsf{pp}, |\mathsf{ct}\rangle, |\psi_\lambda\rangle)\rangle \end{array}\right] = \mathrm{negl}(\lambda).$$

- ***Computational Privacy:*** *Let $\mathsf{Priv\text{-}EXP}^{\mathcal{S}^*}(1^\lambda, \mathcal{M}, t, x)$ be the output of the following experiment.*

  1. *Sample* $(\mathsf{pp}, \mathsf{sp}, \mathsf{vk}, |\mathsf{ct}\rangle) \leftarrow \mathsf{Gen}(1^\lambda, \mathcal{M}, t, x)$.
  2. *Execute* $(b_{\mathsf{OutVer}}, y, \mathsf{C}^*) \leftarrow \mathsf{Eval}\langle \mathcal{C}(\mathsf{sp}), \mathcal{S}^*_\lambda(\mathsf{pp}, |\mathsf{ct}\rangle, |\psi_\lambda\rangle)\rangle$.
  3. *Execute* $(b_{\mathsf{DelVer}}, \mathsf{C}^*) \leftarrow \mathsf{Del}\langle \mathcal{C}(\mathsf{vk}), \mathcal{S}^*_\lambda(b_{\mathsf{OutVer}}, \mathsf{C}^*)\rangle$.
  4. *Run* $\mathcal{S}^*_\lambda(b_{\mathsf{DelVer}}, \mathsf{C}^*)$ *until it outputs a bit.*

  Note that above we have augmented the output of Del *to explicitly include the updated adversary's state on register* C^*. *Computational privacy is satisfied if for any Turing machine $\mathcal{M}$, time bound $t = t(\lambda)$, and sequence of input pairs $\{x_{\lambda,0}, x_{\lambda,1}\}_{\lambda \in \mathbb{N}}$ where each $x_{\lambda,0}$ and $x_{\lambda,1}$ are the same length,*

$$\left| \Pr\left[\mathsf{Priv\text{-}EXP}^{\mathcal{S}^*}(1^\lambda, \mathcal{M}, t, x_{\lambda,0}) = 1\right] - \Pr\left[\mathsf{Priv\text{-}EXP}^{\mathcal{S}^*}(1^\lambda, \mathcal{M}, t, x_{\lambda,1}) = 1\right] \right| = \mathrm{negl}(\lambda).$$

- ***Certified Everlasting Privacy:*** *Let $\mathsf{CD\text{-}EXP}^{\mathcal{S}^*}(1^\lambda, \mathcal{M}, t, x)$ be the output of the following experiment.*

  1. *Sample* $(\mathsf{pp}, \mathsf{sp}, \mathsf{vk}, |\mathsf{ct}\rangle) \leftarrow \mathsf{Gen}(1^\lambda, \mathcal{M}, t, x)$.
  2. *Execute* $(b_{\mathsf{OutVer}}, y, \mathsf{C}^*) \leftarrow \mathsf{Eval}\langle \mathcal{C}(\mathsf{sp}), \mathcal{S}^*_\lambda(\mathsf{pp}, |\mathsf{ct}\rangle, |\psi_\lambda\rangle)\rangle$.
  3. *Execute* $(b_{\mathsf{DelVer}}, \mathsf{C}^*) \leftarrow \mathsf{Del}\langle \mathcal{C}(\mathsf{vk}), \mathcal{S}^*_\lambda(b_{\mathsf{OutVer}}, \mathsf{C}^*)\rangle$.
  4. *If* $b_{\mathsf{DelVer}} = \top$, *output* $(\mathsf{sp}, \mathsf{vk}, \rho)$, *where ρ is the state on register* C^*. *Otherwise, output* $\bot$.

  Certified everlasting privacy is satisfied if for any Turing machine $\mathcal{M}$, time bound $t = t(\lambda)$, and sequence of input pairs $\{x_{0,\lambda}, x_{1,\lambda}\}_{\lambda \in \mathbb{N}}$ of the same length,

$$\mathsf{TD}\left(\mathsf{CD\text{-}EXP}^{\mathsf{S}^*}(1^\lambda, \mathcal{M}, t, x_{\lambda,0}), \mathsf{CD\text{-}EXP}^{\mathsf{S}^*}(1^\lambda, \mathcal{M}, t, x_{\lambda,1})\right) = \mathrm{negl}(\lambda).$$

Definition 3.5 (Publicly-verifiable deletion). *We say that the scheme satisfies* publicly-verifiable deletion *if* pp *includes* vk.

4 Main Theorem

In this section, we consider a generic certified everlasting privacy experiment. This will help us prove security for the full construction of blind delegation later on. Our main theorem is analogous to that of [BK22], but more powerful. We use subspace coset states instead of BB84 states, which ultimately allows us to prove security against a fully-malicious server. Also, we are able to give the adversary an obfuscation of the function that verifies the deletion certificate, which enables schemes with *publicly-verifiable* certified deletion.

The first building block is an encryption scheme $\mathcal{Z}$ with semantic security.

Definition 4.1 (Encryption with semantic security). *For a bit x, let $\{\mathcal{Z}_\lambda(x)\}_{\lambda\in\mathbb{N}}$ be a family of distributions over classical strings or quantum states such that for any class of quantum polynomial-time adversaries $\{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$, it holds that*

$$\Big|\Pr[\mathcal{A}_\lambda(\mathcal{Z}_\lambda(0))=1]-\Pr[\mathcal{A}_\lambda(\mathcal{Z}_\lambda(1))=1]\Big| = \mathrm{negl}(\lambda)$$

For a longer message $x\in\{0,1\}^m$, let $\mathcal{Z}(x)$ denote $\mathcal{Z}_\lambda(x_1),\ldots,\mathcal{Z}_\lambda(x_m)$.

Definition 4.2 (Certified deletion experiment). *For a bit x, let $\mathsf{EXP}_A(1^\lambda,\mathcal{A}_\lambda,\mathcal{Z}_\lambda,x)$ be the output of the following game:*

1. *Challenge: The challenger encrypts x as follows:*
   - *(a) Let $n=4\lambda$. They sample subspaces S,T uniformly at random such that $S\le T$, $\dim(S)=n/2$, and $\dim(T)=3n/4$ using the procedure in Definition 2.2.*
   - *(b) Next they sample vectors $(\mathsf{v},\mathsf{w})\leftarrow \mathsf{co}(S)\times\mathsf{co}(S^\perp)$, and let $\mathsf{u}\in\mathsf{co}(T)$ be the coset of T that v belongs to.*
   - *(c) They send the adversary the following challenge:*

   $$|S_{\mathsf{v},\mathsf{w}}\rangle\,,\mathcal{Z}_\lambda(S,x\oplus\langle\mathsf{v},\mathbf{1}\rangle),T,\mathsf{u}$$

2. *Response: The adversary, running $\mathcal{A}_\lambda$, responds with a deletion certificate $\widetilde{\mathsf{z}}\in\mathbb{F}_2^n$ and an auxiliary state ρ.*

3. *Outcome: The challenger checks that*

   $$\widetilde{\mathsf{z}}\in S^\perp+\mathsf{w}$$

   If so, they output (S,w,ρ); if not, they output $\perp$.

The following theorem says that the outputs of $\mathsf{EXP}_A(1^\lambda,\mathcal{A}_\lambda,\mathcal{Z}_\lambda,0)$ and $\mathsf{EXP}_A(1^\lambda,\mathcal{A}_\lambda,\mathcal{Z}_\lambda,1)$ are statistically close when the adversary is computationally bounded.

Theorem 4.3. *For any class of QPT adversaries $\{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$ and family of distributions $\{\mathcal{Z}_\lambda(x)\}_{\lambda\in\mathbb{N}}$ that satisfy Definition 4.1:*

$$\mathsf{TD}\big[\mathsf{EXP}_A(1^\lambda,\mathcal{A}_\lambda,\mathcal{Z}_\lambda,0),\mathsf{EXP}_A(1^\lambda,\mathcal{A}_\lambda,\mathcal{Z}_\lambda,1)\big]=\mathrm{negl}(\lambda).$$

Assuming the existence of $\mathsf{i}\mathcal{O}$, we can make the deletion certificate publicly verifiable by publishing $\mathsf{i}\mathcal{O}(P_{S^\perp+\mathsf{w}})$. The following experiment, EXP_B, captures this situation.

Theorem 4.4. *Let us assume the existence of post-quantum secure $\mathsf{i}\mathcal{O}$. Next, let EXP_B be the same as EXP_A, except the challenge sent to the adversary also includes $\mathsf{i}\mathcal{O}(P_{S^\perp+\mathsf{w}})$. Then for any class of QPT adversaries $\{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$ and family of distributions $\{\mathcal{Z}_\lambda(x)\}_{\lambda\in\mathbb{N}}$ that satisfy Definition 4.1:*

$$\mathsf{TD}\big[\mathsf{EXP}_B(1^\lambda,\mathcal{A}_\lambda,\mathcal{Z}_\lambda,0),\mathsf{EXP}_B(1^\lambda,\mathcal{A}_\lambda,\mathcal{Z}_\lambda,1)\big]=\mathrm{negl}(\lambda).$$

We will prove Theorem 4.4 below – the proof of Theorem 4.3 is essentially the same.

By immediately combining Theorem 4.4 with constructions in [BK22], we also obtain the following corollary.

Corollary 4.5. *Assuming the existence of post-quantum* $\mathsf{i}\mathcal{O}$ *and post-quantum*

$$X \in \{\textit{public-key encryption}, \textit{attribute-based encryption}, \textit{witness encryption}, \textit{timed-release encryption}\}$$

there exists X *with certified deletion that satisfies publicly-verifiable deletion.*

Proof of Theorem 4.4

Overview: The only part of the challenge that depends on x is the bit $x \oplus \langle \mathsf{v}, \mathbf{1}\rangle$, but here x is masked by a uniformly random bit $\langle \mathsf{v}, \mathbf{1}\rangle$. If the adversary somehow lost all information about v, then x would be information-theoretically hidden as well. Next, note that v and w are encoded in mutually unbiased bases of the same state $|S_{\mathsf{v},\mathsf{w}}\rangle$. Intuitively, an adversary cannot learn both a vector in $S + \mathsf{v}$ and a vector in $S^\perp + \mathsf{w}$ without knowing S. If the adversary's deletion certificate is accepted, then they know a vector $\widetilde{\mathsf{z}} \in S^\perp + \mathsf{w}$, so all information about v is lost.

We'll define a new experiment EXP_C, and then we'll use a hybrid argument to show that if the success probability of EXP_C is negligible (and it is), then the maximum advantage of a QPT adversary in EXP_B is negligible as well. The main difference in the challenge of EXP_C is that it contains $|T_{\mathsf{u},\widetilde{\mathsf{v}}}\rangle$ instead of $|S_{\mathsf{v},\mathsf{w}}\rangle$.

Definition 4.6. *Let* $\mathsf{EXP}_C(1^\lambda, \mathcal{A}_\lambda, \mathcal{Z}_\lambda)$ *be the output of the following game:*

1. *Challenge: The challenger samples* $S, T, \mathsf{u}, \mathsf{w}$ *as they did in* EXP_B. *They also sample* $\widetilde{\mathsf{v}} \leftarrow \mathsf{co}(T^\perp) \cap (S^\perp + \mathsf{w})$. *and* $x' \leftarrow \{0, 1\}$. *Finally, the challenge sent to the adversary is:*
$$|T_{\mathsf{u},\widetilde{\mathsf{v}}}\rangle, \mathcal{Z}_\lambda(S, x'), T, \mathsf{u}, \mathsf{i}\mathcal{O}(P_{S^\perp+\mathsf{w}})$$
2. *Response: Same as in* EXP_B.
3. *Outcome: The challenger checks that*
$$\widetilde{\mathsf{z}} - \widetilde{\mathsf{v}} \in S^\perp \backslash T^\perp$$
The output is 1 *(win) if the check passes and* 0 *(lose) otherwise.*

We'll formalize the argument relating EXP_B to EXP_C with the following hybrids. We'll only state how each hybrid differs from the one before it.

- $\mathsf{Hyb}_0(x)$ is $\mathsf{EXP}_B(1^\lambda, \mathcal{A}_\lambda, \mathcal{Z}_\lambda, x)$.
- $\mathsf{Hyb}_1(x)$: In the *challenge* stage, the challenger samples $x' \leftarrow \{0, 1\}$, and they send the adversary:
$$|S_{\mathsf{v},\mathsf{w}}\rangle, \mathcal{Z}_\lambda(S, x'), T, \mathsf{u}, \mathsf{i}\mathcal{O}(P_{S^\perp+\mathsf{w}})$$
Then in the *outcome* stage, they check whether $x' = x \oplus \langle \mathsf{v}, \mathbf{1}\rangle$. If so, they proceed normally. If not, the game stops, and the outcome is $\perp$.
- $\mathsf{Hyb}_2(x)$: In the *challenge* stage, the challenger samples $S, T, \mathsf{u}, \mathsf{w}$ as before. Then they jointly sample $|S_{\mathsf{v},\mathsf{w}}\rangle$ and v as follows. First, they prepare the state $|\psi\rangle_0$:
$$|\psi\rangle_0 \propto \sum_{\mathsf{v} \in \mathsf{co}(S) \cap (T+\mathsf{u})} |S_{\mathsf{v},\mathsf{w}}\rangle |\mathsf{v}\rangle$$
and then they measure the v register.
- $\mathsf{Hyb}_3(x)$: In the *challenge* stage, the challenger prepares $|\psi\rangle_0$, but they delay measuring v until the start of the *outcome* stage. The challenge still includes the $|S_{\mathsf{v},\mathsf{w}}\rangle$ register, but now it's entangled with the v register.

In the next set of hybrids, we will use a unitary U that depends on the descriptions of S, T, w, and acts on the v register of $|\psi\rangle_0$. For any $\mathsf{v} \in \mathsf{co}(S) \cap (T + \mathsf{u})$:

$$U|\mathsf{v}\rangle = \frac{1}{\sqrt{c}} \sum_{\widetilde{\mathsf{v}} \in \mathsf{co}(T^\perp) \cap (S^\perp + \mathsf{w})} |\widetilde{\mathsf{v}}\rangle \cdot (-1)^{\langle \mathsf{v}, \widetilde{\mathsf{v}} \rangle}$$

where c is a normalization constant. According to Lemma 4.17, when U is applied to the v register of $|\psi\rangle_0$, it maps $|\psi\rangle_0$ to $|\psi\rangle_1$:

$$|\psi\rangle_1 \propto \sum_{\widetilde{\mathsf{v}} \in \mathsf{co}(T^\perp) \cap (S^\perp + \mathsf{w})} |T_{\mathsf{u}, \widetilde{\mathsf{v}}}\rangle |\widetilde{\mathsf{v}}\rangle$$

- Hyb_4: At the start of the *outcome* stage, the challenger applies U to the v register and measures it to get $\widetilde{\mathsf{v}}$. Then they check that
$$\widetilde{\mathsf{z}} - \widetilde{\mathsf{v}} \in S^\perp \backslash T^\perp$$
The output is 1 if the check passes and 0 otherwise.
- Hyb_5: The challenger applies U to the v register and measures the result in the *challenge* stage rather than the *outcome* stage. Therefore the challenge is:
$$|T_{\mathsf{u}, \widetilde{\mathsf{v}}}\rangle, \mathcal{Z}(S, x'), T, \mathsf{u}, \mathsf{i}\mathcal{O}(P_{S^\perp + \mathsf{w}})$$
for a uniformly random $\widetilde{\mathsf{v}} \in \mathsf{co}(T^\perp) \cap (S^\perp + \mathsf{w})$.
- Hyb_6 is EXP_C.

For hybrids Hyb_4 - Hyb_6, let the *value* be the maximum probability, over all QPT adversaries, that the output is 1. We will show that the value is negligible for all three hybrids.

First, Lemma 4.9 shows that the value of Hyb_6 is negligible. Second, Hyb_5 and Hyb_6 have the same value. The only difference between the hybrids is how they sample $(|T_{\mathsf{u}, \widetilde{\mathsf{v}}}\rangle, \widetilde{\mathsf{v}})$, but they still sample from the same distribution.

Third, Hyb_4 and Hyb_5 have the same value. The difference between the hybrids is which of the following operations comes first: (a) the adversary's computation, or (b) the challenger's act of applying U to the v register and measuring it. These two operations commute because they act on disjoint registers: the adversary's input does not include the second register of $|\psi\rangle_0$. Therefore the two hybrids produce identically distributed outcomes.

In Hyb_4, let $\widetilde{\mathsf{v}}' \in \mathsf{co}(T^\perp)$ be the coset of $T^\perp$ that $\widetilde{\mathsf{z}}$ belongs to.

Lemma 4.7. *The adversary loses* Hyb_4 *if and only if* $\widetilde{\mathsf{z}} - \mathsf{w} \notin S^\perp$ *or* $\widetilde{\mathsf{v}} = \widetilde{\mathsf{v}}'$*.*

Proof. If $\widetilde{\mathsf{z}} - \mathsf{w} \notin S^\perp$, then $\widetilde{\mathsf{z}} - \widetilde{\mathsf{v}} \notin S^\perp$ because $\widetilde{\mathsf{v}}$ and w are in the same coset of $S^\perp$, and so the adversary loses. Otherwise, when $\widetilde{\mathsf{z}} - \mathsf{w} \in S^\perp$, the only way to lose Hyb_4 is if $\widetilde{\mathsf{z}} - \widetilde{\mathsf{v}} \in T^\perp$. In this case, $\widetilde{\mathsf{z}}$ and $\widetilde{\mathsf{v}}$ belong to the same coset of $T^\perp$. Since $\widetilde{\mathsf{v}} \in \mathsf{co}(T^\perp)$, $\widetilde{\mathsf{v}} = \widetilde{\mathsf{v}}'$. □

Now, we'll consider Hyb_3. Let its *advantage* be the maximum value of $\mathsf{TD}[\mathsf{Hyb}_3(0), \mathsf{Hyb}_3(1)]$, over all QPT adversaries.

Lemma 4.8. *The advantage of* Hyb_3 *is negligible.*

Proof. If $\widetilde{\mathsf{z}} - \mathsf{w} \notin S^\perp$, then the output of the hybrid is $\perp$. If for some adversary, $\Pr(\widetilde{\mathsf{z}} - \mathsf{w} \in S^\perp)$ is negligible, then the output is $\perp$ with overwhelming probability, and the advantage of Hyb_3 is negligible.

Now take a QPT adversary for which $\Pr(\widetilde{\mathsf{z}} - \mathsf{w} \in S^\perp)$ is non-negligible, and compare how it behaves in Hyb_3 and Hyb_4. The two hybrids have the same *challenge* stage, and we will use the same adversary in the *response* stage, so the state of the system at the start of the *outcome* stage will be the same in both hybrids. At the start of the outcome stage, $\widetilde{\mathsf{z}}, \mathsf{w},$ and T are fixed classical values.

In Hyb_4: conditioned on $\widetilde{\mathsf{z}} - \mathsf{w} \in S^{\perp}$, the adversary loses Hyb_4 with overwhelming probability – otherwise, the adversary would win Hyb_4 with non-negligible probability, contradicting our arguments above. Conditioned on $\widetilde{\mathsf{z}} - \mathsf{w} \in S^{\perp}$, the adversary loses Hyb_4 only if $\widetilde{\mathsf{v}} = \widetilde{\mathsf{v}}\,'$. Therefore, at the start of the outcome stage, conditioned on $\widetilde{\mathsf{z}} - \mathsf{w} \in S^{\perp}$, the state on the v register is within negligible trace distance of $U^{\dagger} |\widetilde{\mathsf{v}}\,'\rangle$.

In Hyb_3: at the start of the *outcome* stage, the state of the system is the same as it was in Hyb_4. But in Hyb_3, the challenger measures the v register without applying U. If the v register has state $U^{\dagger} |\widetilde{\mathsf{v}}\,'\rangle$, then the measured v value is a uniformly random vector in $\mathsf{co}(S) \cap (T + \mathsf{u})$ (Lemma 4.15). Furthermore, if v is uniformly random, then according to Lemma 4.13, $\langle \mathsf{v}, \mathbf{1} \rangle$ is a uniformly random bit (with overwhelming probability over the choice of S, T).

The only part of Hyb_3 that depends on x is when the challenger checks that $x' = x \oplus \langle \mathsf{v}, \mathbf{1} \rangle$. If $\langle \mathsf{v}, \mathbf{1} \rangle$ is uniformly random, then x and $x \oplus \langle \mathsf{v}, \mathbf{1} \rangle$ are independent, and the output of the hybrid is independent of x. This shows that $\mathsf{Advt}(\mathsf{Hyb}_3) = \mathrm{negl}(\lambda)$. □

Finally we'll show that for hybrids Hyb_0 - Hyb_2, the advantage is also negligible. First, Hyb_3 and Hyb_2 have the same advantage because the adversary's computation commutes with the act of measuring v. Second, Hyb_2 and Hyb_1 have the same advantage because the challenge comes from the same distribution in both hybrids – it's just the sampling procedure that's different.

Finally, $\mathsf{Advt}(\mathsf{Hyb}_1) = \frac{1}{2} \cdot \mathsf{Advt}(\mathsf{Hyb}_0)$. In $\mathsf{Hyb}_1(x)$, when $x' = x \oplus \langle \mathsf{v}, \mathbf{1} \rangle$, the challenge and the output of the hybrid have the same distribution as in $\mathsf{Hyb}_0(x)$. When $x' \neq x \oplus \langle \mathsf{v}, \mathbf{1} \rangle$, then the output is $\perp$. x' is independent of x and v, so

$$\mathsf{Advt}(\mathsf{Hyb}_1) = \frac{1}{2} \cdot \mathsf{Advt}(\mathsf{Hyb}_0) + \frac{1}{2} \cdot 0 = \frac{1}{2} \cdot \mathsf{Advt}(\mathsf{Hyb}_0)$$

In summary, $\mathsf{Advt}(\mathsf{Hyb}_0)$ is negligible. Thus, to complete the proof, it suffices to prove the following lemma, which we appealed to earlier to argue that the value of Hyb_6 is negligible.

Lemma 4.9. *For any class of QPT adversaries* $\{\mathcal{A}_\lambda\}_{\lambda \in \mathbb{N}}$*:*

$$\Pr\left[\mathsf{EXP}_C(1^\lambda, \mathcal{A}_\lambda, \mathcal{Z}_\lambda) \to 1\right] = \mathrm{negl}(\lambda)$$

Proof. Intuition: To win EXP_C, the adversary needs more information about S than is provided by T. But to a QPT adversary, any other information that they were given about S is useless. $\mathcal{Z}_\lambda(S, x')$ is unhelpful because S is encrypted, and $\mathsf{i}\mathcal{O}(P_{S^\perp + \mathsf{w}})$ is unhelpful because $\mathsf{i}\mathcal{O}$ is a subspace-hiding obfuscator.

We'll use a sequence of hybrids to show that the value of the first hybrid (EXP_C) cannot be greater than the value of the last hybrid by a non-negligible amount. Then we'll show that the value of the last hybrid is negligible, so the value of EXP_C is negligible as well.

Note that we'll only describe how each hybrid differs from the one before it.

- Hyb_0 is $\mathsf{EXP}_C(1^\lambda, \mathcal{A}_\lambda, \mathcal{Z}_\lambda)$
- Hyb_1 : The challenge is $|T_{\mathsf{u},\widetilde{\mathsf{v}}}\rangle, \mathcal{Z}_\lambda(0^*), T, \mathsf{u}, \mathsf{i}\mathcal{O}(P_{S^\perp + \mathsf{w}})$, where 0^* is the same length as the description of (S, x').
- Hyb_2 : The challenge is $\widetilde{\mathsf{v}}, T, \mathsf{i}\mathcal{O}(P_{S^\perp + \mathsf{w}})$.
- Hyb_3 : Let $\hat{P} \leftarrow \mathsf{i}\mathcal{O}(P_{S^\perp})$. Let $\hat{P}'(\mathsf{z}) = \hat{P}(\mathsf{z} - \mathsf{w})$, for any input $\mathsf{z} \in \mathbb{F}_2^n$, and let $\hat{P}'' \leftarrow \mathsf{i}\mathcal{O}(\hat{P}')$. Finally the challenge is $\widetilde{\mathsf{v}}, T, \hat{P}''$.
- Hyb_4 : The challenge is $\widetilde{\mathsf{v}}, T, \mathsf{i}\mathcal{O}(P_{S^\perp})$.

- Hyb_5 : The challenger samples a subspace R uniformly at random such that $R \leq S$ and $\dim(R) = n/4$. The challenge is $\widetilde{\mathsf{v}}, T, \mathsf{i}\mathcal{O}(P_{R^\perp})$.
- Hyb_6 : The challenge is $\widetilde{\mathsf{v}}, R, T$.

For any QPT adversary, their probability of winning Hyb_0 is negligibly close to that of Hyb_1, by the semantic security (a.k.a CPA security) of $\mathcal{Z}$. If their probability of winning were not negligibly close for the two hybrids, then the $\mathsf{Hyb}_0/\mathsf{Hyb}_1$ adversary could be used to break the CPA security of $\mathcal{Z}$. The CPA adversary would sample $S, T, \mathsf{u}, \widetilde{\mathsf{v}}, \mathsf{w}, x'$ and generate most of the $\mathsf{Hyb}_0/\mathsf{Hyb}_1$ challenge: $|T_{\mathsf{u},\widetilde{\mathsf{v}}}\rangle, T, \mathsf{u}, \mathsf{i}\mathcal{O}(P_{S^\perp+\mathsf{w}})$. Then they would submit (S, x') and 0^* to the CPA challenger and receive $\mathcal{Z}_\lambda(S, x')$ or $\mathcal{Z}_\lambda(0^*)$. They would simulate the hybrid, either Hyb_0 or Hyb_1, using the $\mathsf{Hyb}_0/\mathsf{Hyb}_1$ adversary and output the result of the hybrid. This reduction would yield a CPA adversary with non-negligible advantage.

The success probability of Hyb_1 is less than or equal to that of Hyb_2 because Hyb_2 reduces to Hyb_1. The Hyb_2 adversary can convert their challenge into a Hyb_1 challenge. They would sample $\mathcal{Z}(0^*)$ and u and use $(\mathsf{u}, \widetilde{\mathsf{v}}, T)$ to compute $|T_{\mathsf{u},\widetilde{\mathsf{v}}}\rangle$. Finally, any response that wins in the simulation of Hyb_1 would also win in Hyb_2. Therefore the success probability of Hyb_2 is at least as large as that of Hyb_1.

For any QPT adversary, their probability of winning Hyb_2 is negligibly close to that of Hyb_3, by the security of $\mathsf{i}\mathcal{O}$. $\hat{P}'$ is functionally equivalent to $P_{S^\perp+\mathsf{w}}$, so their obfuscations, $\hat{P}''$ and $\mathsf{i}\mathcal{O}(P_{S^\perp+\mathsf{w}})$ respectively, are indistinguishable to any QPT adversary. If the success probabilities of Hyb_2 and Hyb_3 were not negligibly close, then an $\mathsf{i}\mathcal{O}$ adversary could use the $\mathsf{Hyb}_2/\mathsf{Hyb}_3$ adversary to break the security of $\mathsf{i}\mathcal{O}$. The $\mathsf{i}\mathcal{O}$ adversary would sample $S, T, \mathsf{u}, \widetilde{\mathsf{v}}, \mathsf{w}, x'$ and then submit $\hat{P}'$ and $P_{S^\perp+\mathsf{w}}$ to the $\mathsf{i}\mathcal{O}$ challenger and receive an obfuscation, either $\hat{P}''$ or $\mathsf{i}\mathcal{O}(P_{S^\perp+\mathsf{w}})$. Then they would generate the challenge for Hyb_2 or Hyb_3 and run the $\mathsf{Hyb}_2/\mathsf{Hyb}_3$ adversary on the challenge. They would simulate the rest of the hybrid and output the hybrid's outcome. This adversary would have a non-negligible distinguishing advantage in the $\mathsf{i}\mathcal{O}$ security game.

The success probability of Hyb_3 is less than or equal to that of Hyb_4 because Hyb_4 reduces to Hyb_3. Given a Hyb_4 challenge, the Hyb_4 adversary can convert it to a Hyb_3 challenge by computing $\hat{P}''$ from $\mathsf{i}\mathcal{O}(P_{S^\perp})$. Through this reduction, any Hyb_3 adversary would yield a Hyb_4 adversary with the same probability of winning.

For any QPT adversary, their probability of winning Hyb_4 is negligibly close to that of Hyb_5 because $\mathsf{i}\mathcal{O}$ is a subspace-hiding obfuscator (Theorem 2.5). That means that $\mathsf{i}\mathcal{O}(P_{S^\perp})$ is indistinguishable from $\mathsf{i}\mathcal{O}(P_{R^\perp})$ for any QPT adversary. Note that $R^\perp$ is a random superspace of $S^\perp$ of dimension $3n/4$.

The success probability of Hyb_5 is less than or equal to that of Hyb_6 because Hyb_6 reduces to Hyb_5. Given a Hyb_6 challenge, the Hyb_6 adversary can convert it to a Hyb_5 challenge by computing $\mathsf{i}\mathcal{O}(P_{R^\perp})$.

In summary, we've shown that the success probability of Hyb_0 is at most negligibly greater than that of Hyb_6. To finish the proof, we'll show that the success probability of Hyb_6 is negligible, so the success probability of Hyb_0 is negligible as well.

Lemma 4.10. *Any adversary has negligible probability of winning* Hyb_6.

Proof. The adversary receives $(\widetilde{\mathsf{v}}, R, T)$, where R and T are uniformly random subspaces such that $\dim(R) = n/4$, $\dim(T) = 3n/4$, and $R \leq T$. Also $\widetilde{\mathsf{v}} \leftarrow \mathsf{co}(T^\perp)$. Then they output $\widetilde{\mathsf{z}}$, and they win if $\widetilde{\mathsf{z}} - \widetilde{\mathsf{v}} \in S^\perp \backslash T^\perp$. However, $S^\perp$ is unknown to the adversary, so it's impossible for them to find a vector in $S^\perp \backslash T^\perp$ with non-negligible probability.

For any given $(\widetilde{\mathsf{v}}, R, T, \widetilde{\mathsf{z}})$, $S^\perp$ is a uniformly random subspace s.t. $T^\perp \leq S^\perp \leq R^\perp$ and $\dim(S^\perp) = n/2$. The probability that a given $(\widetilde{\mathsf{v}}, R, T, \widetilde{\mathsf{z}})$ wins Hyb_0 is

$$\begin{aligned}\Pr_S\left[\widetilde{\mathsf{z}} - \widetilde{\mathsf{v}} \in S^\perp \backslash T^\perp \,\middle|\, \widetilde{\mathsf{v}}, R, T, \widetilde{\mathsf{z}}\right] &\leq \max_{\mathsf{z} \in \mathbb{F}_2^n} \Pr_S\left(\mathsf{z} \in S^\perp \backslash T^\perp\right) \\ &= \max_{\mathsf{z} \in R^\perp \backslash T^\perp} \Pr_S\left(\mathsf{z} \in S^\perp \backslash T^\perp\right) = \frac{|S^\perp \backslash T^\perp|}{|R^\perp \backslash T^\perp|} = \frac{2^{n/2} - 2^{n/4}}{2^{3n/4} - 2^{n/4}} = \mathrm{negl}(\lambda)\end{aligned}$$

Here we used the fact that $\dim(T^\perp) = n/4$, $\dim(S^\perp) = n/2$, and $\dim(R^\perp) = 3n/4$. □

□

Auxiliary lemmas about subspaces

Definition 4.11.

- *Let* $S, T \leq \mathbb{F}_2^n$ *be sampled according to Definition 2.2 such that* $\dim(S) = n/2, \dim(T) = 3n/4$, *and* $S \leq T$.
- *Let* $\mathsf{u} \in \mathsf{co}(T)$ *and* $\mathsf{w} \in \mathsf{co}(S^\perp)$ *be chosen arbitrarily.*
- *Let* $A = \mathsf{co}(S) \cap (T + \mathsf{u})$ *and* $B = \mathsf{co}(T^\perp) \cap (S^\perp + \mathsf{w})$
- *Let* $c = |T|/|S|$

Lemma 4.12.

$$|A| = |B| = c$$

Proof. Since S is a subgroup of T, each coset of T can be partitioned into c cosets of S. $T + \mathsf{u}$ is a coset of T, so $|\mathsf{co}(S) \cap (T + \mathsf{u})| = c$.

Next, $T^\perp \leq S^\perp$ because $S \leq T$. Therefore $S^\perp + \mathsf{w}$ can be partitioned into $|S^\perp|/|T^\perp|$ cosets of $T^\perp$. Finally,

$$|\mathsf{co}(T^\perp) \cap (S^\perp + \mathsf{w})| = \frac{|S^\perp|}{|T^\perp|} = \frac{2^n}{|S|} \cdot \frac{|T|}{2^n} = c$$

□

Lemma 4.13. *With overwhelming probability in* n *over the randomness of* S *and* T, *exactly half of the vectors* $\mathsf{v} \in A$ *satisfy* $\langle \mathsf{v}, \mathbf{1} \rangle = 1$, *and the other half satisfy* $\langle \mathsf{v}, \mathbf{1} \rangle = 0$.

Proof. Over the randomness of S and T, $\mathsf{co}(S) \cap T$ is a uniformly random subspace of dimension $n/4$. With overwhelming probability, this subspace contains at least one vector z^* with parity 1. Half of the vectors in $\mathbb{F}_2^n$ have parity 1, so the probability that all the vectors in $\mathsf{co}(S) \cap T$ have parity 0 is $\leq (1/2)^{n/4} = \mathrm{negl}(n)$.

If there is at least one $\mathsf{z}^* \in \mathsf{co}(S) \cap T$ with parity 1, then exactly half of the vectors in $\mathsf{co}(S) \cap T$ have parity 1. This is because $\mathsf{co}(S) \cap T$ is a subspace. For any vector $\mathsf{z} \in \mathsf{co}(S) \cap T$, the parity of $\mathsf{z}' = \mathsf{z} + \mathsf{z}^*$ is different from the parity of z. z and z' are both in $\mathsf{co}(S) \cap T$, so half of the vectors in $\mathsf{co}(S) \cap T$ have parity 1 and half have parity 0.

Next, every $\mathsf{v} \in A$ can be written as $\mathsf{v} = \mathsf{z} + \mathsf{u}$ for a unique $\mathsf{z} \in \mathsf{co}(S) \cap T$. This is because A is a coset of $\mathsf{co}(S) \cap T$. $\mathsf{u} \in \mathsf{co}(T) \leq \mathsf{co}(S)$, so $(\mathsf{co}(S) \cap T) + \mathsf{u} = \mathsf{co}(S) \cap (T + \mathsf{u}) = A$.

Next, the parity of v can be written in terms of the parity of z:

$$\langle \mathsf{v}, \mathbf{1} \rangle = \langle \mathsf{z}, \mathbf{1} \rangle + \langle \mathsf{u}, \mathbf{1} \rangle$$

$\langle \mathsf{u}, \mathbf{1} \rangle$ is constant, so if half of the z-vectors have parity 1 and the other half have parity 0, then the same is true of the vectors $\mathsf{v} \in A$: exactly half of the vectors have parity 1, and the rest have parity 0. □

Definition 4.14.

- *Let* $\overline{A} = \mathbb{F}_2^n \backslash A$, *and* $\overline{B} = \mathbb{F}_2^n \backslash B$.
- *Let* f *be a bijective mapping from* $\overline{A}$ *to* $\overline{B}$.
- *Let* U *be the linear function mapping any* $\mathsf{v} \in A$ *to*

$$U |\mathsf{v}\rangle = \frac{1}{\sqrt{c}} \sum_{\widetilde{\mathsf{v}} \in B} |\widetilde{\mathsf{v}}\rangle \cdot (-1)^{\langle \mathsf{v}, \widetilde{\mathsf{v}} \rangle}$$

  and any $\mathsf{v} \in \overline{A}$ *to* $|f(\mathsf{v})\rangle$.

- *Let* $|\psi\rangle_0 = \frac{1}{\sqrt{c}} \sum_{\mathsf{v} \in A} |S_{\mathsf{v},\mathsf{w}}\rangle |\mathsf{v}\rangle$

- *Let* $|\psi\rangle_1 = \frac{1}{\sqrt{c}} \sum_{\widetilde{\mathsf{v}} \in B} |T_{\mathsf{u},\widetilde{\mathsf{v}}}\rangle |\widetilde{\mathsf{v}}\rangle$

There always exists a bijective mapping f from $\overline{A}$ to $\overline{B}$ because both sets have size $2^n - c$ (Lemma 4.12).

Lemma 4.15. *$U^\dagger$ is the linear function mapping any $\widetilde{\mathsf{v}} \in B$ to*

$$U^\dagger |\widetilde{\mathsf{v}}\rangle = \frac{1}{\sqrt{c}} \sum_{\mathsf{v} \in A} |\mathsf{v}\rangle \cdot (-1)^{\langle \mathsf{v}, \widetilde{\mathsf{v}} \rangle}$$

and any $\widetilde{\mathsf{v}} \in \overline{B}$ to $|f^{-1}(\widetilde{\mathsf{v}})\rangle$.

Proof. For any $\mathsf{v} \in A$ and any $\widetilde{\mathsf{v}} \in B$,

$$\langle \widetilde{\mathsf{v}} | U | \mathsf{v} \rangle = \frac{1}{\sqrt{c}} \cdot (-1)^{\langle \mathsf{v}, \widetilde{\mathsf{v}} \rangle}$$

If $\mathsf{v} \notin A$ or $\widetilde{\mathsf{v}} \notin B$, then

$$\langle \widetilde{\mathsf{v}} | U | \mathsf{v} \rangle = \mathbb{1}_{\widetilde{\mathsf{v}} == f(\mathsf{v})}$$

□

Lemma 4.16. *U is unitary.*

Proof. To prove the claim, we will show that for any $\mathsf{v}, \mathsf{v}' \in \mathbb{F}_2^n$, $\langle \mathsf{v}' | U^\dagger U | \mathsf{v} \rangle = \mathbb{1}_{\mathsf{v} == \mathsf{v}'}$.

First, if v or v' are not in A, then

$$\langle \mathsf{v}' | U^\dagger U | \mathsf{v} \rangle = \mathbb{1}_{\mathsf{v}' == f^{-1} \circ f(\mathsf{v})} = \mathbb{1}_{\mathsf{v} == \mathsf{v}'}$$

Now let $\mathsf{v}, \mathsf{v}' \in A$.

$$\begin{aligned}
\langle \mathsf{v}' | U^\dagger U | \mathsf{v} \rangle &= \frac{1}{c} \sum_{\widetilde{\mathsf{v}}' \in B} \sum_{\widetilde{\mathsf{v}} \in B} \langle \widetilde{\mathsf{v}}' | \widetilde{\mathsf{v}} \rangle \cdot (-1)^{\langle \mathsf{v}, \widetilde{\mathsf{v}} \rangle - \langle \mathsf{v}', \widetilde{\mathsf{v}}' \rangle} \\
&= \frac{1}{c} \sum_{\widetilde{\mathsf{v}} \in B} (-1)^{\langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}} \rangle}
\end{aligned}$$

If $\mathsf{v} = \mathsf{v}'$, then $\langle \mathsf{v}' | U^\dagger U | \mathsf{v} \rangle = \frac{1}{c} \sum_{\widetilde{\mathsf{v}} \in B} 1 = 1$.

Now let $\mathsf{v} - \mathsf{v}' \neq 0$. Lemma 4.20 says that for half of the vectors $\widetilde{\mathsf{v}} \in B$, $\langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}} \rangle = 1$, and for the other half, $\langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}} \rangle = 0$. Therefore,

$$\langle \mathsf{v}' | U^\dagger U | \mathsf{v} \rangle = \frac{1}{c} \sum_{\widetilde{\mathsf{v}} \in B} (-1)^{\langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}} \rangle} = \frac{1}{2} - \frac{1}{2} = 0$$

We've shown that for any $\mathsf{v}, \mathsf{v}' \in \mathbb{F}_2^n$, $\langle \mathsf{v}' | U^\dagger U | \mathsf{v} \rangle = \mathbb{1}_{\mathsf{v} == \mathsf{v}'}$, so U is unitary. □

Lemma 4.17. *Applying U to the v register of $|\psi\rangle_0$ produces $|\psi\rangle_1$:*

$$\frac{1}{\sqrt{c}} \sum_{\mathsf{v} \in A} |S_{\mathsf{v},\mathsf{w}}\rangle \otimes (U |\mathsf{v}\rangle) = \frac{1}{\sqrt{c}} \sum_{\widetilde{\mathsf{v}} \in B} |T_{\mathsf{u},\widetilde{\mathsf{v}}}\rangle |\widetilde{\mathsf{v}}\rangle$$

Proof.

$$\text{LHS} = \frac{1}{\sqrt{c \cdot |S|}} \sum_{\mathsf{v} \in A} \sum_{\mathsf{z} \in S + \mathsf{v}} |\mathsf{z}\rangle \otimes (U |\mathsf{v}\rangle) \cdot (-1)^{\langle \mathsf{z}, \mathsf{w} \rangle} \tag{1}$$

$$= \frac{1}{\sqrt{c \cdot |S|}} \sum_{\mathsf{z} \in T + \mathsf{u}} \sum_{\mathsf{v} \in A \,:\, \mathsf{z} - \mathsf{v} \in S} |\mathsf{z}\rangle \otimes (U |\mathsf{v}\rangle) \cdot (-1)^{\langle \mathsf{z}, \mathsf{w} \rangle} \tag{2}$$

$$= \frac{1}{\sqrt{c \cdot |S|}} \sum_{\mathsf{z} \in T + \mathsf{u}} |\mathsf{z}\rangle \otimes (U |\mathsf{v}\rangle) \cdot (-1)^{\langle \mathsf{z}, \mathsf{w} \rangle} \quad \text{where } \mathsf{v} \in \mathsf{co}(S) \text{ is } \mathsf{z}\text{'s coset of } S \tag{3}$$

For Eq. (2) we changed the order of summation. z takes every value in $\mathsf{span}(S, A)$, which equals $T + \mathsf{u}$ (Lemma 4.21). Furthermore, z takes each value only once because $A \subseteq \mathsf{co}(S)$, for every z, there is a unique $(\mathsf{s}, \mathsf{v}) \in S \times \mathsf{co}(S)$ such that $\mathsf{z} = \mathsf{s} + \mathsf{v}$ (Lemma 2.1). For Eq. (3), we removed the summation over v because v is uniquely determined by z.

$$\begin{aligned}
\text{LHS} &= \frac{1}{\sqrt{c^2 \cdot |S|}} \sum_{\widetilde{\mathsf{v}} \in B} \sum_{\mathsf{z} \in T + \mathsf{u}} |\mathsf{z}\rangle\, |\widetilde{\mathsf{v}}\rangle \cdot (-1)^{\langle \mathsf{z}, \mathsf{w}\rangle + \langle \mathsf{v}, \widetilde{\mathsf{v}}\rangle} && (4)\\
&= \frac{1}{\sqrt{c^2 \cdot |S|}} \sum_{\widetilde{\mathsf{v}} \in B} \sum_{\mathsf{z} \in T + \mathsf{u}} |\mathsf{z}\rangle\, |\widetilde{\mathsf{v}}\rangle \cdot (-1)^{\langle \mathsf{z}, \widetilde{\mathsf{v}}\rangle} && (5)\\
&= \sqrt{\frac{|T|}{c^2 \cdot |S|}} \sum_{\widetilde{\mathsf{v}} \in B} |T_{\mathsf{u}, \widetilde{\mathsf{v}}}\rangle\, |\widetilde{\mathsf{v}}\rangle = \frac{1}{\sqrt{c}} \sum_{\widetilde{\mathsf{v}} \in B} |T_{\mathsf{u}, \widetilde{\mathsf{v}}}\rangle\, |\widetilde{\mathsf{v}}\rangle && (6)\\
&= \text{RHS} && (7)
\end{aligned}$$

For Eq. (5), we used the fact that $\langle \mathsf{z}, \widetilde{\mathsf{v}}\rangle = \langle \mathsf{z}, \mathsf{w}\rangle + \langle \mathsf{v}, \widetilde{\mathsf{v}}\rangle$ (Lemma 4.22). □

Lemma 4.18. *$A - \mathsf{u} = \mathsf{co}(S) \cap T$ and $B - \mathsf{w} = \mathsf{co}(T^\perp) \cap S^\perp$.*

Proof. First, $A = \mathsf{co}(S) \cap (T + \mathsf{u})$. Next, $\mathsf{u} \in \mathsf{co}(T) \le \mathsf{co}(S)$, so $A - \mathsf{u} \subseteq \mathsf{co}(S) \cap T$. Finally, $|A| = |\mathsf{co}(S) \cap T|$, so $A = \mathsf{co}(S) \cap T$.

Second, $B = \mathsf{co}(T^\perp) \cap (S^\perp + \mathsf{w})$. Next, $\mathsf{w} \in \mathsf{co}(S^\perp) \le \mathsf{co}(T^\perp)$ (Lemma 2.3), so $B - \mathsf{w} \subseteq \mathsf{co}(T^\perp) \cap S^\perp$. Finally, $|B| = |\mathsf{co}(T^\perp) \cap S^\perp|$, so $B = \mathsf{co}(T^\perp) \cap S^\perp$. □

Lemma 4.19. $(A - \mathsf{u}) \cap (B - \mathsf{w})^\perp = \{\mathbf{0}\}$

Proof. First, $S^\perp \cap \mathsf{co}(S^\perp) = \{\mathbf{0}\}$ because $S^\perp$ shares exactly one vector with $\mathsf{co}(S^\perp)$ (Lemma 2.1). It has to be $\mathbf{0}$ because $S^\perp$ and $\mathsf{co}(S^\perp)$ are subspaces. By the same logic, $T^\perp \cap \mathsf{co}(T^\perp) = \{\mathbf{0}\}$.

Second, $[\mathsf{co}(T^\perp) \cap S^\perp]^\perp = [\mathsf{co}(T)^\perp \cap S^\perp]^\perp = \mathsf{span}[\mathsf{co}(T), S]$. Next, let z be some vector in $\mathsf{co}(S) \cap T \cap [\mathsf{co}(T^\perp) \cap S^\perp]^\perp$. z can be decomposed into two vectors $(\mathsf{z}', \mathsf{z}'') \in \mathsf{co}(T) \times S$ such that $\mathsf{z} = \mathsf{z}' + \mathsf{z}''$.

Third, $\mathsf{z} \in T$ and $\mathsf{z}'' \in S \le T$, so $\mathsf{z}' = \mathsf{z} - \mathsf{z}'' \in T$. But $\mathsf{z}' \in \mathsf{co}(T)$ as well, so $\mathsf{z}' = \mathbf{0}$. Therefore, $\mathsf{z} = \mathsf{z}'' \in S$. But $\mathsf{z} \in \mathsf{co}(S)$ as well, so $\mathsf{z} = \mathbf{0}$. Therefore, $\mathsf{co}(S) \cap T \cap [\mathsf{co}(T^\perp) \cap S^\perp]^\perp = \{\mathbf{0}\}$.

□

Lemma 4.20. *For any $\mathsf{v}, \mathsf{v}' \in A$, if $\mathsf{v} \neq \mathsf{v}'$, then for exactly half of the vectors $\widetilde{\mathsf{v}} \in B$, $\langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}}\rangle = 1$, and for the other half, $\langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}}\rangle = 0$.*

Proof. First, note that $\mathsf{v} - \mathsf{v}' \in A - \mathsf{u}$ because $A - \mathsf{u}$ is a subspace, and v, v' belong to the same coset of it. Second, $\langle \mathsf{v} - \mathsf{v}', \mathsf{w}\rangle = 0$ because $\mathsf{w} \in \mathsf{co}(S^\perp) = \mathsf{co}(S)^\perp$ (Definition 2.2), and $A - \mathsf{u} \subseteq \mathsf{co}(S)$.

Third, for any non-zero $\mathsf{v} - \mathsf{v}' \in (A - \mathsf{u})$, there exists a $\widetilde{\mathsf{v}}^* \in B$ for which $\langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}}^* - \mathsf{w}\rangle = 1$. Otherwise, $\mathsf{v} - \mathsf{v}' \in (B - \mathsf{w})^\perp$, which is ruled out by Lemma 4.19.

Fourth, for any $\widetilde{\mathsf{v}} \in B$, let $\widetilde{\mathsf{v}}' = \widetilde{\mathsf{v}} + \widetilde{\mathsf{v}}^* - \mathsf{w}$. Note that $\widetilde{\mathsf{v}}' \in B$ because $B - \mathsf{w}$ is a subspace, and

$$\widetilde{\mathsf{v}}' - \mathsf{w} = (\widetilde{\mathsf{v}} - \mathsf{w}) + (\widetilde{\mathsf{v}}^* - \mathsf{w}) \in B - \mathsf{w}$$

Fifth:

$$\begin{aligned}
\langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}}' - \mathsf{w}\rangle &= \langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}} - \mathsf{w}\rangle + \langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}}^* - \mathsf{w}\rangle\\
&\neq \langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}} - \mathsf{w}\rangle\\
\langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}}'\rangle &\neq \langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}}\rangle
\end{aligned}$$

We've shown that B can be partitioned into pairs of vectors $(\widetilde{\mathsf{v}}, \widetilde{\mathsf{v}}')$ such that $\langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}}'\rangle \neq \langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}}\rangle$. Therefore, for exactly half of the vectors $\widetilde{\mathsf{v}} \in B$, $\langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}}\rangle = 1$, and for the other half, $\langle \mathsf{v} - \mathsf{v}', \widetilde{\mathsf{v}}\rangle = 0$. □

Lemma 4.21. $\mathsf{span}(S, A) = T + \mathsf{u}$.

Proof. First, $\mathsf{span}(S, A) \subseteq T + \mathsf{u}$ because $S \subseteq T$, and $A \subseteq T + \mathsf{u}$. Furthermore, for every $\mathsf{z} \in T + \mathsf{u}$, there is a pair $(\mathsf{z}', \mathsf{z}'') \in S \times \mathsf{co}(S)$ such that $\mathsf{z} = \mathsf{z}' + \mathsf{z}''$ (Lemma 2.1).

Next, $\mathsf{z}'' \in A$ because $\mathsf{z}'' \in \mathsf{co}(S)$ and $\mathsf{z}'' \in T + \mathsf{u}$. To see this, observe that $\mathsf{z}'' = \mathsf{z} - \mathsf{z}' \in \mathsf{span}(T + \mathsf{u}, S) = T + \mathsf{u}$. Then $\mathsf{z} \in \mathsf{span}(S, A)$, so $T + \mathsf{u} \subseteq \mathsf{span}(S, A)$, and in fact $\mathsf{span}(S, A) = T + \mathsf{u}$. □

Lemma 4.22. *Let* $\mathsf{v} \in \mathsf{co}(S)$, $\widetilde{\mathsf{v}} \in B$, $\mathsf{w} \in \mathsf{co}(S^\perp)$, *and* $\mathsf{z} \in S + \mathsf{v}$. *Then* $\langle \mathsf{z}, \widetilde{\mathsf{v}} \rangle = \langle \mathsf{z}, \mathsf{w} \rangle + \langle \mathsf{v}, \widetilde{\mathsf{v}} \rangle$.

Proof. First, $\mathsf{z} - \mathsf{v} \in S$, and $\widetilde{\mathsf{v}} - \mathsf{w} \in S^\perp$, so $\langle \mathsf{z} - \mathsf{v}, \widetilde{\mathsf{v}} - \mathsf{w} \rangle = 0$. Second, $\mathsf{w} \in \mathsf{co}(S^\perp) = \mathsf{co}(S)^\perp$ (Definition 2.2), so $\langle \mathsf{v}, \mathsf{w} \rangle = 0$. Then:

$$\begin{aligned} \langle \mathsf{z} - \mathsf{v}, \widetilde{\mathsf{v}} - \mathsf{w} \rangle &= 0 \\ \langle \mathsf{z}, \widetilde{\mathsf{v}} \rangle &= \langle \mathsf{z}, \mathsf{w} \rangle + \langle \mathsf{v}, \widetilde{\mathsf{v}} \rangle - \langle \mathsf{v}, \mathsf{w} \rangle \\ \langle \mathsf{z}, \widetilde{\mathsf{v}} \rangle &= \langle \mathsf{z}, \mathsf{w} \rangle + \langle \mathsf{v}, \widetilde{\mathsf{v}} \rangle \end{aligned}$$

□

5 Construction of Blind Delegation with Certified Deletion

In this section, we'll construct a fully-secure blind delegation with certified deletion scheme. Here is the construction at a high level: The client encodes x in a quantum ciphertext of the form $\mathsf{C} = (\mathsf{Z}, \mathsf{ct}_0)$. Z is a quantum register, and ct_0 is a classical FHE ciphertext. Then the server computes $\mathcal{M}_t$ homomorphically on C. Next the client decrypts the server's response to learn $y = \mathcal{M}_t(x)$. The server also proves that they did the computation correctly using a SNARG, which the client verifies.

However, the SNARG only certifies that the mapping from input to output was correctly computed. The client must also verify that the input Z has the correct value. This is handled in Eval step 2b. Some care is required here to prevent the client's measurement on Z from changing the state and thereby giving an adversarial server useful information.

Finally, to delete the ciphertext, the client and server uncompute their previous computations; then the server measures Z in the Hadamard basis, which destroys the information needed to recover x. The value that the server measured serves as a certificate of deletion, which the client verifies.

Now here is the construction in detail:

$\underline{\mathsf{Gen}}(1^\lambda, \mathcal{M}, t, x)$:

1. The client sets up an FHE scheme:
$$\mathsf{FHE.Gen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$$
2. They encrypt x:
   First let m be the length of x, and let $n = 4\lambda$. Then,
   (a) For each $i \in [m]$, they sample a subspace S_i of dimension $n/2$, along with two vectors $(\mathsf{v}_i, \mathsf{w}_i) \leftarrow \mathsf{co}(S_i) \times \mathsf{co}(S_i^\perp)$, uniformly at random. Then they compute:
$$\begin{aligned} x_i' &= x_i \oplus \langle \mathsf{v}_i, \mathbf{1} \rangle \\ \mathsf{Z}_i &= |(S_i)_{\mathsf{v}_i, \mathsf{w}_i}\rangle \end{aligned}$$
   Z_i is a quantum register, and together the Z_is constitute the register $\mathsf{Z} := (\mathsf{Z}_i)_{i \in [m]}$.
   (b) For each $i \in [m]$, they sample a subspace T_i such that $S_i \leq T_i$ and $\dim(T_i) = 3n/4$, from the procedure specified in Definition 2.2. Next, let $\mathsf{u}_i \in \mathsf{co}(T_i)$ be the coset of T_i that v_i belongs to.

(c) Then they generate the ciphertext on register C:

$$\mathsf{ct}_0 = \mathsf{FHE.Enc}\Big[\mathsf{pk}, \big(S_i, x_i'\big)_{i\in[m]}\Big]$$
$$\mathsf{C} = (\mathsf{Z}, \mathsf{ct}_0)$$

3. They set up a SNARG scheme for the Turing machine $\mathcal{M}' = \mathsf{HEval}(\mathsf{pk}, \mathcal{M}, t, \cdot, \mathsf{ct}_0)$. ($\mathcal{M}'$ takes a classical string from register Z as input and computes $\mathcal{M}_t(x)$ homomorphically. It is defined later). To do so, they compute $l = \mathrm{poly}(\lambda, m)$, which upper bounds the input length of $\mathcal{M}'$, along with $t' = \mathrm{poly}(\lambda, m, t)$, which upper bounds the runtime of $\mathcal{M}'$. Then they compute:

$$\mathsf{SNARG.Gen}(1^\lambda, \mathcal{M}', t', l) \to (\mathsf{SNARG\_pk}, \mathsf{SNARG\_vk})$$

4. They compile the existing information to make $\mathsf{pp}, \mathsf{sp}, \mathsf{vk}$:

$$\mathsf{pp} = \mathcal{M}, t, \mathsf{pk}, \mathsf{SNARG\_pk}$$
$$\mathsf{sp} = \mathsf{pp}, \mathsf{sk}, \mathsf{SNARG\_vk}, (T_i, \mathsf{u}_i)_{i\in[m]}$$
$$\mathsf{vk} = (S_i, \mathsf{w}_i)_{i\in[m]}$$

5. **Outputs**: $(\mathsf{C}, \mathsf{pp})$ are sent to the server, and the client keeps $(\mathsf{sp}, \mathsf{vk})$.

Eval:

1. Server:

   (a) The server applies $\mathcal{M}' = \mathsf{HEval}(\mathsf{pk}, \mathcal{M}, t, \cdot, \mathsf{ct}_0)$ in superposition to Z, while computing a SNARG certifying that the computation was done correctly. With our notation, the description of the functionality $\mathcal{M}'$ is determined by the public key $\mathsf{SNARG\_pk}$, so we simply write

   $$\mathsf{Y}, \mathsf{P} \leftarrow \mathsf{SNARG.Prove}\big(\mathsf{SNARG\_pk}, \mathsf{Z}\big),$$

   where Y is the register containing the output of HEval, and P is the register containing the SNARG proof. They are entangled with each other and with Z.

   (b) The server sends registers $\mathsf{Z}, \mathsf{Y}, \mathsf{P}$ to the client.

2. Client:

   (a) The client verifies the SNARG. First, they compute in superposition the function:

   $$\mathsf{SNARG.Verify}\big(\mathsf{SNARG\_vk}, \mathsf{Z}, \mathsf{Y}, \mathsf{P}\big)$$

   and then they measure the output bit. If the output is 1, they continue. Otherwise, they **output** $(y = \bot, b_{\mathsf{OutVer}} = \bot)$ and skip to step 2d.

   (b) Next, they verify the input Z. For each $i \in [m]$, they check that $\mathsf{Z}_i \in T_i + \mathsf{u}_i$ (this involves a measurement). If every check passes, they continue. Otherwise, they **output** $(y = \bot, b_{\mathsf{OutVer}} = \bot)$ and skip to step 2d.

   (c) The client decrypts Y in superposition, by running $\mathsf{FHE.Dec}(\mathsf{sk}, \mathsf{Y})$, and measures the result. Let y be the measured value, and let $b_{\mathsf{OutVer}} = \top$. Finally, the client **outputs** (y, b_{OutVer}).

   (d) The client uncomputes the computations they performed (except measurements) and sends the registers $\mathsf{Z}, \mathsf{Y}, \mathsf{P}$ back to the server.

3. Server: The server uncomputes the computations they performed. The last **output** is register C.

Del:

1. Server:
    (a) For each $i \in [m]$, the server applies $H^{\otimes n}$ to Z_i and then measures the register to get value $\widetilde{\mathsf{z}}_i$.
    (b) They send the client the deletion certificate $\mathsf{cert} := (\widetilde{\mathsf{z}}_i)_{i\in[m]}$.
2. Client:
    (a) For each $i \in [m]$, the client checks that $\widetilde{\mathsf{z}}_i \in S_i^{\perp} + \mathsf{w}_i$. If all checks pass, they **output** $b_{\mathsf{DelVer}} = \top$, and otherwise they **output** $b_{\mathsf{DelVer}} = \perp$.

All that's left to do is define HEval and a helper function, $\mathsf{UnmaskInput}$. HEval computes $\mathcal{M}_t(x)$ homomorphically. But the input to HEval requires preprocessing. The message corresponding to ct_0 contains a masked version of x:

$$(x_i \oplus \langle \mathsf{v}_i, \mathbf{1}\rangle)_{i\in[m]}$$

The purpose of $\mathsf{UnmaskInput}$ is to remove the mask from x so that it can be given as input to $\mathcal{M}_t$.

Finally, HEval and $\mathsf{UnmaskInput}$ are classical functions that the server applies in superposition. Therefore, we will express their inputs as classical values. For instance, we will use z_i to represent a generic classical value in register Z_i's superposition.

The output of $\mathsf{UnmaskInput}$ is a string x'', which equals x when the server and client are honest.

$\underline{\mathsf{UnmaskInput}}\big[(\mathsf{z}_i)_{i\in[m]}, (S_i, x'_i)_{i\in[m]}\big]$:

1. For each $i \in [m]$:
    (a) Compute the coset $\mathsf{v}'_i \in \mathsf{co}(S_i)$ that z_i belongs to.
    (b) Compute $x''_i = x'_i \oplus \langle \mathsf{v}'_i, \mathbf{1}\rangle$.
2. **Output** $x'' := (x''_1, \ldots, x''_m)$.

The output of HEval is an encryption of $\mathcal{M}_t(x)$ when the server and client are honest.

$\underline{\mathsf{HEval}}\big[\mathsf{pk}, \mathcal{M}, t, (\mathsf{z}_i)_{i\in[m]}, \mathsf{ct}_0\big]$:

1. Encrypt the z_is

$$\mathsf{ct}_1 = \mathsf{FHE.Enc}\big[\mathsf{pk}, (\mathsf{z}_i)_{i\in[m]}\big]$$

2. Homomorphically compute $\mathsf{UnmaskInput}$

$$\mathsf{ct}_2 = \mathsf{FHE.Eval}\big[\mathsf{pk}, \mathsf{UnmaskInput}, (\mathsf{ct}_1, \mathsf{ct}_0)\big]$$

3. Homomorphically compute $\mathcal{M}_t$. First, let $C_{\mathcal{M}_t}$ be the circuit description of $\mathcal{M}_t$. Then compute:

$$\mathsf{ct}_3 = \mathsf{FHE.Eval}\big(\mathsf{pk}, C_{\mathcal{M}_t}, \mathsf{ct}_2\big)$$

Finally, **output** ct_3.

5.1 Efficiency and Correctness

Theorem 5.1. *The construction in Section 5 of blind delegation with certified deletion is efficient (Definition 3.2) if $\mathcal{M}$ has a description of size $\mathrm{poly}(|x|, \log t, \lambda)$.*

Proof. As before, let $m = |x|$.

First, the client runs in time $\mathrm{poly}(m, \log t, \lambda)$. The client runs several helper functions: $\mathsf{FHE.Gen}$ takes time $\mathrm{poly}(\lambda)$, $\mathsf{SNARG.Gen}$ takes time $\mathrm{poly}(m, \log t, \lambda)$, and $\mathsf{SNARG.Verify}$ takes time $\mathrm{poly}(m, \log t, \lambda)$. It is easy to verify that the rest of the client's computations also take time $\mathrm{poly}(m, \log t, \lambda)$.

Next, the server runs in time $\mathrm{poly}(m, t, \lambda)$. The longest step is running $\mathsf{SNARG.Prove}$, which takes time $\mathrm{poly}(m, t, \lambda)$. The entire blind delegation protocol, and in particular the server's computation, takes time $\mathrm{poly}(m, t, \lambda)$ as well. □

Theorem 5.2. *The construction in Section 5 is correct (Definition 3.3).*

Proof. We will show that with overwhelming probability in λ: $y = \mathcal{M}_t(x)$, $b_{\mathsf{OutVer}} = \top$, and $b_{\mathsf{DelVer}} = \top$.

At the start of Eval, every Z_i holds a superposition of classical values z_i that satisfy $\mathsf{z}_i \in S_i + \mathsf{v}_i$. First imagine using the z_is to compute:

$$\mathsf{UnmaskInput}\big((\mathsf{z}_i)_{i\in[m]}, (S_i, x'_i)_{i\in[m]}\big)$$

For each i: $\mathsf{v}'_i = \mathsf{v}_i$, and that means $x''_i = x_i \oplus \langle \mathsf{v}_i, \mathbf{1}\rangle \oplus \langle \mathsf{v}'_i, \mathbf{1}\rangle = x_i$, so the output of $\mathsf{UnmaskInput}$ would be x.

Next, imagine using the z_is to compute:

$$\mathcal{M}'[(z_i)_{i\in[m]}] = \mathsf{HEval}[\mathsf{pk}, \mathcal{M}, t, (\mathsf{z}_i)_{i\in[m]}, \mathsf{ct}_0]$$

HEval encrypts the z_is and applies $\mathsf{UnmaskInput}$ homomorphically, so the result, ct_2, is an encryption of x. Next, $\mathcal{M}_t$ (in ciruit form) is applied homomorphically to ct_2, so the result, ct_3, is an encryption of $\mathcal{M}_t(x)$, and this is the output of HEval.

Now we will show that the outcome of Eval is $[y = \mathcal{M}_t(x), b_{\mathsf{OutVer}} = \top]$ with overwhelming probability. In Eval step 1a, $\mathsf{HEval}(\mathsf{pk}, \mathcal{M}, t, \cdot, \mathsf{ct}_0)$ is applied in superposition to Z, and the result on register Y is a superposition of encryptions of $\mathcal{M}_t(x)$. In Eval steps 2a-b, the client performs some tests, which pass with certainty. First, $\mathsf{SNARG.Verify}$ passes, by the correctness of the SNARG. Second, the checks in step 2b pass because for each i: $\mathsf{z}_i \in S + \mathsf{v}_i \subseteq T_i + \mathsf{u}_i$. In step 2c of Eval, the client decrypts Y and measures the result. Y decrypts to $y = \mathcal{M}_t(x)$, by the correctness of FHE. Also, $b_{\mathsf{OutVer}} = \top$.

The measurements in Eval step 2 produce a particular outcome with overwhelming probability, so the measurements change the state of the quantum registers by a negligible amount (measured in trace distance). When the client and server uncompute their computations in Eval steps 2d and 3, the final state on each Z_i is negligibly close to what it was originally: $\mathsf{Z}_i = |(S_i)_{\mathsf{v}_i,\mathsf{w}_i}\rangle$.

Finally, we'll show that $b_{\mathsf{DelVer}} = \top$ with overwhelming probability. In Del step 1a, $H^{\otimes n}\mathsf{Z}_i = |(S_i^\perp)_{\mathsf{w}_i,\mathsf{v}_i}\rangle$. Measuring the result produces $\widetilde{\mathsf{z}}_i \in S_i^\perp + \mathsf{w}_i$. Then in step 2, the client verifies the $\widetilde{\mathsf{z}}_i$ values, and all the checks pass, so $b_{\mathsf{DelVer}} = \top$. □

5.2 Security

In this section, we prove security for our construction. We recall that $P_{S_i+\mathsf{v}_i}(\mathsf{Z}_i)$ checks that $\mathsf{Z}_i \in S_i + \mathsf{v}_i$.

Lemma 5.3. *If the checks in* Eval *step 2b pass, then after that step, every* Z_i *is within negligible trace distance of a state for which* $\Pr[P_{S_i+\mathsf{v}_i}(\mathsf{Z}_i)$ *accepts*$] = 1$.

Proof. We can view the part of the blind delegation protocol that ends after Eval step 2b as a game. The adversarial server will try to produce a register Z_i for which $P_{T_i+\mathsf{u}_i}(\mathsf{Z}_i)$ accepts but $P_{S_i+\mathsf{v}_i}(\mathsf{Z}_i)$ rejects. Since T_i is a uniformly random superspace of S_i (of dimension $3n/4$), winning the game is information-theoretically impossible.

To be more formal, the game is the following: Let the adversary control the server, and let the challenger implement the client. The two parties follow the blind delegation protocol until the end of Eval step 2a. Then for every i, the challenger checks that check that $\mathsf{Z}_i \in T_i + \mathsf{u}_i \backslash S_i + \mathsf{v}_i$. The adversary wins if *any* of the checks pass, and they lose otherwise.

From the server's point of view, each T_i is a uniformly random subspace such that $T_i \geq S_i$ and $\dim(T_i) = 3n/4$. Therefore, the probability that the challenger's test accepts on a given register i is:

$$\Pr[P_{T_i+\mathsf{u}_i\backslash S_i+\mathsf{v}_i}(\mathsf{Z}_i) \text{ accepts}] \leq \max_{\mathsf{z}\in\mathbb{F}_2^n} \Pr_{T_i}[\mathsf{z} \in T_i + \mathsf{u}_i\backslash S_i + \mathsf{v}_i] = \operatorname{negl}(\lambda)$$

where the probability is taken over all T_i such that $\dim(T_i) = 3n/4$ and $T_i \geq S_i$. The probability that the adversary wins the game is $\leq m \cdot \operatorname{negl}(\lambda) = \operatorname{negl}(\lambda)$. □

Theorem 5.4. *The construction in Section 5 has integrity (Definition 3.4).*

Proof. We want to show that for every QPT adversarial server, $\Pr[b_{\mathsf{OutVer}} = \top \wedge y \neq \mathcal{M}_t(x_\lambda)]$ is negligible. First, if for some adversary, $\Pr[b_{\mathsf{OutVer}} = \top]$ is negligible, then the condition is satisfied. Now assume that for a given adversary, $\Pr[b_{\mathsf{OutVer}} = \top]$ is non-negligible. Then we will show that $\Pr[y = \mathcal{M}_t(x_\lambda) | b_{\mathsf{OutVer}} = \top]$ is overwhelming.

First, if $b_{\mathsf{OutVer}} = \top$, then $\mathsf{SNARG.Verify}$ passed in Eval step 2a. Since $b_{\mathsf{OutVer}} = \top$ with non-negligible probability, $\mathsf{SNARG.Verify}$ also passes with non-negligible probability. Given that $\mathsf{SNARG.Verify}$ passed, then with overwhelming probability, the value on Y is the result of running $\mathsf{HEval}(\mathsf{pk}, \mathcal{M}, t, \mathsf{Z}, \mathsf{ct}_0)$ (Definition 2.6).

Second, if b_{OutVer} passed, then the checks in Eval step 2b also passed. By Lemma 5.3, each Z_i is negligibly close in trace distance to a state for which $\Pr[P_{S_i+\mathsf{v}_i}(\mathsf{Z}_i) \text{ accepts}] = 1$.

Third, if every $\mathsf{Z}_i \in S_i + \mathsf{v}_i$ and HEval was correctly computed, then Y is a superposition of encryptions of $\mathcal{M}_t(x)$, so the client will recover $y = \mathcal{M}_t(x)$ with certainty. Therefore, if $b_{\mathsf{OutVer}} = \top$, then the client gets $y = \mathcal{M}_t(x)$ with overwhelming probability.

□

Corollary 5.5. *If the measurement in step 2c yields $y = \mathcal{M}_t(x)$, then the quantum state before and after the measurement are negligibly close in trace distance.*

Proof. In the proof of Theorem 5.4, we showed that if Eval step 2c is executed, the value measured in that step is $y = \mathcal{M}_t(x)$ with overwhelming probability. This is a gentle measurement, so it changes the state of the system by a negligible amount – the state before and after the measurement are negligibly close in trace distance. □

Theorem 5.6. *The construction in Section 5 has computational privacy (Definition 3.4).*

Proof. Corollary 5.5 says that the measurement in Eval step 2c changes the quantum state by a negligible amount. Eval step 2c is the only place where the decryption key sk is used. Since that step changes the state negligibly, it is as if the server's inputs are independent of sk. Then computational privacy follows from the CPA security of $\mathsf{FHE.Enc}$.

We'll make this formal with a sequence of hybrids. Note that we'll only say how each hybrid differs from the one before it:

- Hyb_0 is $\mathsf{Priv\text{-}EXP}^{\mathcal{S}^*}(1^\lambda, \mathcal{M}, t, x)$, the computational privacy security game.
- Hyb_1: In step 2c, the challenger does not touch the quantum state. Instead, they compute $\mathcal{M}_t(x)$ on their own and output $y = \mathcal{M}_t(x)$ and $b_{\mathsf{OutVer}} = \top$.

- Hyb_2: In Gen step 2c, let $\mathsf{ct}_0 = \mathsf{FHE.Enc}(0^*)$, where 0^* is the same length as $\left(S_i, x'_i\right)_{i\in[m]}$.

The output distributions of Hyb_0 and Hyb_1 are negligibly close because the quantum state after Eval step 2c is negligibly close in trace distance between the two hybrids. Next, the output distributions of Hyb_1 and Hyb_2 are negligibly close by the semantic security of $\mathsf{FHE.Enc}$. Finally, in Hyb_3, the output is independent of x because y is the only variable that depends on x, but the server does not get access to y. Therefore, the adversary's advantage is 0 in Hyb_3. In summary, the adversary's advantage in $\mathsf{Priv\text{-}EXP}$ is negligible. □

Theorem 5.7. *The construction in Section 5 has certified everlasting privacy (Definition 3.4).*

Proof. We will show that for any λ and any distinct inputs $(x_{\lambda,0}, x_{\lambda,1})$ of the same length, an adversarial server cannot produce a deletion certificate that is accepted, and at the same time distinguish the two inputs with more than negligible advantage.

First consider the case where $(x_{\lambda,0}, x_{\lambda,1})$ differ on just one bit (without loss of generality, let them differ on the first bit). In this case, we can prove certified everlasting privacy with a reduction to Game 4.2. Assume that there is a QPT adversarial server that breaks certified everlasting privacy. Then there is an adversary for Game 4.2 that simulates the certified everlasting privacy game $\mathsf{CD\text{-}EXP}$ (Definition 3.4) and achieves non-negligible advantage in Game 4.2.

Let's fix some notation. The *adversary* and *challenger* are the players in Game 4.2, and the *server* and *client* are the players in $\mathsf{CD\text{-}EXP}$. The input to $\mathsf{CD\text{-}EXP}$ is a bitstring $x \in \{x_{\lambda,0}, x_{\lambda,1}\}$ of length m. The first bit, x_1, may be 0 or 1, but the remaining bits are fixed. The input to Game 4.2 will be just the first bit, x_1. Next, in Game 4.2, $\mathcal{Z}_\lambda$ is a distribution representing any encryption scheme with semantic security. For concreteness, let $\mathcal{Z}_\lambda(x) = [\mathsf{FHE.Enc}(\mathsf{pk}, x), \mathsf{pk}]$ for some $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda)$ computed by the challenger.

Now here is what Game 4.2 looks like with the adversary that simulates $\mathsf{CD\text{-}EXP}$:

1. The challenger sets up the encryption scheme:

$$\mathsf{FHE.Gen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$$

Then they sample subspaces (S_1, T_1) and vectors $(\mathsf{u}_1, \mathsf{v}_1, \mathsf{w}_1)$ from the specified distribution. Then they compute $x'_1 = x_1 \oplus \langle \mathsf{v}_1, 1\rangle$. Finally they send the adversary

$$|(S_1)_{\mathsf{v}_1,\mathsf{w}_1}\rangle, \mathsf{FHE.Enc}[\mathsf{pk}, (S_1, x'_1)], \mathsf{pk}, T_1, \mathsf{u}_1$$

2. (a) The adversary simulates the rest of Gen step 2. For every $i \in \{2, \ldots, m\}$, they sample $(S_i, T_i, \mathsf{u}_i, \mathsf{v}_i, \mathsf{w}_i)$. They also compute x'_i from x_i, which is fixed. Finally, they put everything together to produce C.

   (b) They run Gen step 3, which entails setting up the SNARG scheme. For Gen step 4, the adversary computes pp. However they don't compute sp or vk because that would require knowing $(\mathsf{sk}, S_1, \mathsf{w}_1)$.

   (c) The adversary simulates Eval by executing the client and server's algorithms, except instead of running step 2c, they simply output $(y = 0, b_{\mathsf{OutVer}} = \top)$ (because step 2c step requires sk).

   (d) The adversary simulates Del step 1, in which the server outputs a deletion certificate $(\widetilde{\mathsf{z}}_i)_{i\in[m]}$ and a state ρ. Then the adversary sends the challenger the first vector $\widetilde{\mathsf{z}}_1$ along with the state

$$\rho' = \left[\rho, \mathcal{M}, t, \mathsf{pk}, \mathsf{SNARG\_pk}, \mathsf{SNARG\_vk}, T_1, \mathsf{u}_1, (S_i, T_i, \mathsf{u}_i, \mathsf{w}_i)_{i\geq 2}\right]$$

3. The challenger checks whether $\widetilde{z}_1 \in S_1^\perp + \mathsf{w}_1$. If so they output $(S_1, \mathsf{w}_1, \rho')$ and if not, they output $\perp$.

The procedure above implements the blind delegation protocol as written, except for Gen step 4 and Eval step 2c, because those steps require $(\mathsf{sk}, S_1, \mathsf{w}_1)$, which the adversary does not have. However, the procedure still simulates the server's view correctly. In Gen step 4, the adversary does not compute sp or vk, but since the server does not receive sp or vk in the blind delegation protocol, the adversary doesn't need to compute them to simulate the server's view. Also, after Eval step 2c is executed, the server receives the quantum registers that were acted on as well as b_{OutVer}, but they don't receive y. b_{OutVer} always equals $\top$ when 2c is executed, and step 2c changes the state of the registers by a negligible amount (Corollary 5.5). Therefore the adversary simulates the server's view of Eval step 2c (up to a negligible trace distance).

If the server can break certified everlasting privacy in CD-EXP, then with non-negligible probability, $b_{\mathsf{DelVer}} = \top$. When this occurs, the output of CD-EXP is (sp, ρ), and the advantage, $\mathsf{TD}[(\mathsf{sp}, \rho)_0, (\mathsf{sp}, \rho)_1]$, is non-negligible. In the simulation of CD-EXP, the outcome is essentially the same. With non-negligible probability, the challenger's check passes in step 3 of Game 4.2, and in that case, the output of the game is $(S_1, \mathsf{w}_1, \rho')$.

Next, $\mathsf{TD}[(\mathsf{sk}, S_1, \mathsf{w}_1, \rho')_0, (\mathsf{sk}, S_1, \mathsf{w}_1, \rho')_1]$ is non-negligible because $(\mathsf{sk}, S_1, \mathsf{w}_1, \rho')$ contains all of $(\mathsf{sp}, \mathsf{vk}, \rho)$. But even without sk, the advantage: $\mathsf{TD}[(S_1, \mathsf{w}_1, \rho')_0, (S_1, \mathsf{w}_1, \rho')_1]$ is still non-negligible. Recall that trace distance, like statistical distance, is the maximum distinguishing advantage of an unbounded quantum adversary. Given $(S_1, \mathsf{w}_1, \rho')$, an adversary with unbounded time can sample a valid sk from the same distribution as the one used by the challenger. Using pk, they prepare a list of all sks that are valid companions to pk, and they sample one of them. Therefore the adversary's distinguishing advantage cannot decrease when we remove sk:

$$\mathsf{TD}[(S_1, \mathsf{w}_1, \rho')_0, (S_1, \mathsf{w}_1, \rho')_1] = \mathsf{TD}[(\mathsf{sk}, S_1, \mathsf{w}_1, \rho')_0, (\mathsf{sk}, S_1, \mathsf{w}_1, \rho')_1]$$

This contradicts Theorem 4.3, which says that $\mathsf{TD}[(S_1, \mathsf{w}_1, \rho)_0, (S_1, \mathsf{w}_1, \rho)_1]$ must be negligible. Therefore the blind delegation protocol does in fact satisfy certified everlasting privacy for any QPT adversary.

Now we'll generalize this proof to work for any distinct $x_{\lambda,0}$ and $x_{\lambda,1}$ of the same length, even when they differ on multiple bits. We'll use a set of hybrids going from $x_{\lambda,0}$ to $x_{\lambda,1}$ in which adjacent hybrids differ by a single bit:

- Hyb_0: Run CD-EXP with input $x_{\lambda,0}$.

For any $0 < i < m$:

- Hyb_i : Let x^i be the bitstring of length m where the first $m - i$ bits match $x_{\lambda,0}$ and the remaining bits match $x_{\lambda,1}$. Then run CD-EXP with input x^i.
- Hyb_m: Run CD-EXP with input $x_{\lambda,1}$.

For any adjacent hybrids $(\mathsf{Hyb}_i, \mathsf{Hyb}_{i+1})$, the inputs (x^i, x^{i+1}) differ on exactly one bit. Therefore $\mathsf{TD}(\mathsf{Hyb}_i, \mathsf{Hyb}_{i+1})$ is negligible. Next,

$$\mathsf{TD}(\mathsf{Hyb}_0, \mathsf{Hyb}_m) \leq \sum_{i=0}^{m-1} \mathsf{TD}(\mathsf{Hyb}_i, \mathsf{Hyb}_{i+1}) = \mathrm{negl}(\lambda) \cdot m = \mathrm{negl}(\lambda)$$

This shows that certified everlasting privacy is satisfied for any $(x_{\lambda,0}, x_{\lambda,1})$ of the same length. $\square$

Remark 5.8. *To conclude this section, we remark that if we replace* $\mathsf{vk} = (S_i, \mathsf{w}_i)_{i\in[m]}$ *with* $\mathsf{vk} = \{\mathsf{i}\mathcal{O}(P_{S_i^\perp + \mathsf{w}_i})\}_{i\in[m]}$*, then assuming that* $\mathsf{i}\mathcal{O}$ *is a post-quantum secure indistinguishability obfuscator, we achieve blind delegation with certified deletion that satisfies* publicly-verifiable deletion *(Definition 3.5). The proof is the same, except that we appeal to Theorem 4.4 rather than Theorem 4.3 in the proof of certified everlasting privacy.*

References

[AKL+] Prabhanjan Ananth, Fatih Kaleoglu, Xingjian Li, Qipeng Liu, and Mark Zhandry. On the feasibility of unclonable encryption, and more. CRYPTO 2022.

[BDGM22] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Factoring and Pairings Are Not Necessary for IO: Circular-Secure LWE Suffices. In Mikołaj Bojańczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022)*, volume 229 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 28:1–28:20, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[BF10] Niek J. Bouman and Serge Fehr. Sampling in a quantum population, and applications. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 724–741. Springer, Heidelberg, August 2010.

[BGMZ18] James Bartusek, Jiaxin Guan, Fermi Ma, and Mark Zhandry. Return of GGH15: Provable security against zeroizing attacks. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part II*, volume 11240 of *LNCS*, pages 544–574. Springer, Heidelberg, November 2018.

[BI20] Anne Broadbent and Rabib Islam. Quantum encryption with certified deletion. LNCS, pages 92–122. Springer, Heidelberg, March 2020.

[BK22] James Bartusek and Dakshita Khurana. Cryptography with certified deletion. Cryptology ePrint Archive, Paper 2022/1178, 2022. https://eprint.iacr.org/2022/1178.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 97–106, 2011.

[CJJ21] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. Snargs for $\mathcal{P}$ from LWE. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 68–79. IEEE, 2021.

[CLLZ21] Andrea Coladangelo, Jiahui Liu, Qipeng Liu, and Mark Zhandry. Hidden cosets and applications to unclonable cryptography. In *Advances in Cryptology – CRYPTO 2021: 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16–20, 2021, Proceedings, Part I*, page 556–584, Berlin, Heidelberg, 2021. Springer-Verlag.

[CVW18] Yilei Chen, Vinod Vaikuntanathan, and Hoeteck Wee. GGH15 beyond permutation branching programs: Proofs, attacks, and candidates. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 577–607. Springer, Heidelberg, August 2018.

[DQV$^+$21] Lalita Devadas, Willy Quach, Vinod Vaikuntanathan, Hoeteck Wee, and Daniel Wichs. Succinct LWE sampling, random polynomials, and obfuscation. LNCS, pages 256–287. Springer, Heidelberg, 2021.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, page 169–178, New York, NY, USA, 2009. Association for Computing Machinery.

[GP21] Romain Gay and Rafael Pass. Indistinguishability obfuscation from circular security. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, page 736–749, New York, NY, USA, 2021. Association for Computing Machinery.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 75–92. Springer, Heidelberg, August 2013.

[HMNY21] Taiga Hiroka, Tomoyuki Morimae, Ryo Nishimaki, and Takashi Yamakawa. Quantum encryption with certified deletion, revisited: Public key, attribute-based, and classical communication. LNCS, pages 606–636. Springer, Heidelberg, 2021.

[HMNY22a] Taiga Hiroka, Tomoyuki Morimae, Ryo Nishimaki, and Takashi Yamakawa. Certified everlasting functional encryption. *CoRR*, abs/2207.13878, 2022.

[HMNY22b] Taiga Hiroka, Tomoyuki Morimae, Ryo Nishimaki, and Takashi Yamakawa. Certified everlasting zero-knowledge proof for QMA, 2022. Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, Santa Barbara, CA, USA.

[NSBU16] Daniel Nagaj, Or Sattath, Aharon Brodutch, and Dominique Unruh. An adaptive attack on wiesner's quantum money. *Quantum Info. Comput.*, 16(11–12):1048–1070, sep 2016.

[Por22] Alexander Poremba. Quantum proofs of deletion for learning with errors. Cryptology ePrint Archive, Report 2022/295, 2022.

[VZ21] Thomas Vidick and Tina Zhang. Classical proofs of quantum knowledge. LNCS, pages 630–660. Springer, Heidelberg, 2021.

[WW21] Hoeteck Wee and Daniel Wichs. Candidate obfuscation via oblivious LWE sampling. LNCS, pages 127–156. Springer, Heidelberg, 2021.

[Zha19] Mark Zhandry. Quantum lightning never strikes the same state twice. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 408–438. Springer, Heidelberg, May 2019.